

中央财经大学学术著作
出版资助基金资助出版

仿射括号代数理论算法与应用

FANGSHE KUOHAO DAISHU LILUN SUANFA YU YINGYONG

张　宁◎著

知识产权出版社

责任编辑 / 兰　涛
封面设计 / 張冀書裝 WWW.8088K.COM



张　宁男，1978年生，辽宁朝阳人。2006年毕业于中国科学院数学院，获博士学位，现任中央财经大学中国精算研究院专职研究人员。在《中国科学》等杂志上发表论文数篇，部分论文被SCI/EI索引。研究方向为：经济学中的基础数学理论、金融保险中的模拟方法及最优策略等。讲授的课程为：风险统计模型、破产理论等。

上架建议：数学科学类



ISBN 978-7-80247-309-6/O・003
（10224）定　价：18.00元

仿射括号代数理论
算法与应用

张 宁 著

知识产权出版社

内容提要

本书介绍了括号代数的基本理论、仿射括号代数的相关定义与定理以及用于表示的一些相关技术，并在此基础上给出了一些相应的算法，同时附有系统实现的主算法、系统实现的的相关细节以及 Maplc 程序的编制技术。本书适用于高等院校数学系、计算机系的自动推理、不变量应用等专业的高年级本科生及研究生阅读。

责任编辑：兰　涛

图书在版编目(CIP)数据

仿射括号代数理论算法与应用/张宁著. —北京:知识产权出版社,2008.6

ISBN 978-7-80247-309-6

Ⅰ.仿…　Ⅱ.张…　Ⅲ.仿射—代数—算法　Ⅳ.018

中国版本图书馆 CIP 数据核字(2008)第 080311 号

仿射括号代数理论算法与应用

张　宁　著

出版发行：知识产权出版社

社　　址：	北京市海淀区马甸南村 1 号	邮　　编：	100088
网　　址：	http://www.ipph.cn	邮　　箱：	bjb@cnipr.com
发行电话：	010-82000893　82000860 转 8101	传　　真：	010-82000860-8325
责编电话：	010-82000860-8325	责编邮箱：	lantao@cnipr.com
印　　刷：	知识产权出版社电子制印中心	经　　销：	新华书店及相关销售网点
开　　本：	880mm×1230mm　1/32	印　　张：	5.5
版　　次：	2008 年 6 月第一版	印　　次：	2009年2月第2次印刷
字　　数：	115 千字	定　　价：	18.00 元

ISBN　978-7-80247-309-6/O·003(10224)

衷心感谢

感谢中央财经大学“学术著作出版资助基金”的资助！

感谢本书的编辑兰涛女士！

衷心感谢我的恩师和师兄师姐、师弟师妹们！

感谢中国精算研究院的全体同事：孙宝文教授、李晓林教授、齐玲教授、寇业富副教授、刘达副教授、林光彬副研究员以及周明、高洪忠、董洪斌、韩光华、姚海波、欧阳和霞、黄敏燕、李石保、薛丽娜等老师，感谢他们给予我的帮助！

谨以此书献给我的父母、我的妻子蒋彬以及我可爱的女儿，感谢他们！

前　言

括号代数作为不变量理论的一种基本工具已经被广泛研究，并且展示了它的巨大优势。但是相关的研究成果都是以外文形式存在，本书将对此给出一个简要的介绍和说明。同时作为括号代数的自然扩展，仿射括号代数的相关研究很少。无论是它的有效算法还是它的具体应用都极少有人研究，所以本书的主要内容集中于该领域的研究。这些内容主要综合了笔者以前的一些研究成果以及后期发表的论文，同时参考了师友的相关工作。本书在这些相关主题之间进行了必要的衔接过渡，以期通俗易懂、逐步深入。

本书主要内容如下：不变量理论的基本介绍；括号代数理论的基本介绍；仿射括号代数与括号代数的整除性理论；仿射括号代数的展开理论；仿射括号代数的交换算法以及其他算法；仿射括号代数在“自动证明”等方面的应用等。

本书内容的特点体现在以下方面。

1. 讨论了包含边界算子的展开问题，分析了边界算子一些新的性质，给出了新的相关重要公式；完善和补充了仿射几

何的构造方式、元素的表示以及消元方法。

2. 提出了交换算法等几个有效的仿射括号代数新算法。

3. 讨论了括号代数、仿射括号代数的整除性问题。

4. 给出了仿射括号代数以及括号代数的基本运算系统的实现方式。

5. 基于符号计算软件 Maple 10 中实现了仿射括号代数的基本算法，并对此系统做了详细的介绍。

6. 书中附有大量的实例以及关键程序代码。

目　录

第1章 引　言

本章介绍了不变量理论及括号代数的历史发展，同时分析了进行该研究的必要性。仿射括号代数作为括号代数的年轻分支，无论从理论上还是应用上都有很大的研究空间。本章最后特别说明了本书研究的主要具体问题、主要内容以及主要结果。

1.1　括号代数发展及应用

不变量理论是数学发展史上一个年轻而又富有活力的领域，而括号代数正是它的有效工具之一。

我们知道最早的解析几何本质上是研究矩阵、向量在一些线性群的作用下不变性质的学科，而括号代数正是这种思想应用于射影几何的结果。

1890 年和 1893 年，Hilbert 运用不变量理论首次证明了

Hilbert 三大定理：零点定理、基底定理和合冲理想定理[13~15]。它们为现代代数的发展奠定了基础。通常我们从代数的角度出发认为，不变量理论研究的是在某些群作用下多项式的不变性质。Klein 从几何角度出发，认为不变量理论是研究在几何变化下几何图形的不变性质。所以不变量理论其实是与坐标无关的，那么是否有一种研究不变量的无坐标工具呢？1926 年，Van der Waerden 运用括号给出了不变量理论中的恒等式。1936 年，Weitzenböck 给出了 Grassmann-Plücker 关系式生成特征为零的多重向量空间上的理想的结论，即特征为零的不变量第二基本定理。在此之后，1939 年 Weyl 给出了特征为零的域上的不变量理论第二基本定理：f是 Grassmann 代数簇生成理想中的一个多项式，如果 f 在行列式是 1 的线性群的作用下不变，则f可以被表示为括号生成的多项式。

不变量理论和括号代数接着经历了 20 多年的沉睡期，并在 20 世纪 70 年代重新繁荣起来。1971 年，Whilteley 从逻辑角度证明了所有的射影几何定理都可以用括号来表述和证明。1973 年 White 系统地给出了括号代数的定义。之后，分别在 1974 年和 1976 年，Doubilet，Rota 和 Stein，Concini，Procesi 给出了不变量理论第一基本定理在任意特征的无限域上都是成立的基本理论[71,72]。1978 年 Désarměnien，Kunng 和 Rota 给出了不变量的第二基本定理在任意特征的无限域上都是成立的基本理论。1989 年，Sturmfels 和 White 把 Alfred Young 的拉直算法应用于括号代数，并且给出了括号代数的合冲理想的 Gröbner 基，解决了括号代数在数域上的表示问题。

需要说明的是随着括号代数的相关研究的进展，它也被应用到越来越多的领域。

第一类是括号代数在射影几何中的应用：

（1）对射影几何模型的括号表示基本完善。最基本的几何表示是根据几何意义得到超平面的几何表示，三点共线的几何表示以及交比的表示。二次曲线的表示是由 White 在 1989 年给出的，在此，射影平面的二次曲线可以表示为 4 次 2 项的括号方程，同样他也给出了射影空间中二次曲面的表示，这是一个 5 次 138 项的括号方程。1991 年，Whiteley 给出了一般的几何模型都可以用括号坐标来得到括号多项式表示的结论，当然这里不是最小表示，比用坐标多项式表示还要复杂。

（2）寻找坐标表示的多项式的几何意义。假定要寻找一个坐标形式的多项式f的几何意义，首先需要把f转化为括号多项式，然后进行 Cayley 分解，从而可以得到f的几何意义，但是一般来说第一个过程因为有 syzygy 的存在而变得非常困难。对于第二步，1988 年，White 和 Mcmillan 对此做出了一些初步的工作。1991 年 White 给出了多重线性括号多项式的 Cayley 分解算法。同年 Sturmfels 等给出了整系数的齐次括号多项式的 Cayley 分解算法。

第二类是括号代数在离散几何方面的应用：

（1）格论不等式的表达。1999～2000 年，Mainetti 和 Yan 运用 Grassmann-Cayley 代数、括号代数给出了格论不等式的表示，并且发现这些不等式对应着射影几何定理。

（2）抽象图的坐标化。可计算综合几何的一个基本问题

就是找到抽象定义的图的坐标化或者不可实现的证明。括号代数和 Grassmann-Cayley 代数可以很好地解决这类问题。在过去的 20 年，这方面已经取得了长足的进步。

第三类是括号代数在代数几何上的应用：例如用于 Chow 形的计算，1994 年 Dalbec 和 Sturmfels 运用 Grassmann 坐标极大地简化了 Chow 形的计算。

第四类是括号代数在计算机视觉（computer vision）中的应用。

1991 年，Crapo 运用不变量理论给出了三维场景恢复的条件。1995 年，Faugeras 和 Mourrain 运用 Grassmann-Cayley 代数以及括号代数给出了匹配图像之间点和线对应的约束。1999 年，Bayro-Corrochano 等运用括号形式的 Passcal 定理模型对摄像机标定（calibration），同年，Csurka 等运用括号计算了两幅图像之间的不变量，并且应用到图像匹配方面。2004 年之后，Hongbo Li 和 Lina Zhao 等首次运用括号代数、共形几何代数进行了线画图的高维恢复。

第五类是括号代数在机械学、结构学以及机器人学方面的应用：这当中包括运用括号代数分析连杆机构的奇异性条件、刚体的无限小描述等，也包括著名的 Steward 平台的相关研究等。

第六类是括号代数在计算机图形学、物理学等方面的应用：例如曲线曲面造型，宇宙学等相关方面。

第七类是括号代数在自动证明以及自动推理领域的应用。

1.2 自动证明与括号代数的应用

数学通常的形式有两种，一种是烦琐但是可以机械化的运算，另外一种是技巧性稍高的证明。王浩先生对比了计算和证明：计算容易、烦琐、刻板和枯燥；而证明困难、简略、灵活而优美。那么是不是证明也可以机械化呢？这就是自动证明问题。

早在17世纪，人们就已经有这样的想法，这条思路的根本就是把证明这种高技巧的脑力劳动转化为虽然烦琐但是可以刻板化有一定通用性的计算。直到19世纪末，在一批数学家的努力下，这种方法才有了明确的数学刻画，而真正的转折点是在计算机出现之后，这是因为：烦琐的计算需要计算机来支持，而机械化的思想和计算的共性可以用程序来实现，这样自动证明就蓬勃发展起来了。

20世纪初期，数理逻辑学家深入探讨了定理自动证明的可能性，得出的结论大多是悲观的。例如歌德尔就曾提出，即使是初等数论中的定理，进行自动证明也是不可能的。同时，有数学家从另外一个角度提出，初等几何定理进行自动证明是可能的。

美国于1956年开始进行机器自动证明的研究。1959年王浩先生用机械化的算法证明了《数学原理》中数百条定理，

引起了轰动，但是之后的很多研究却遭遇了挫折。这时我国著名数学家吴文俊先生提出了吴方法，这在定理自动证明领域取得了巨大的成功。

进行自动证明的方法基本上分为两大类，一类是坐标化方法，例如著名的吴方法；另外一类是无坐标的定理自动证明方法。无坐标的方法大致有以下几种。

（1）第一种方法是括号代数。

1991 年 Whiteley 利用括号坐标表示一个几何模型，并从 Hilbert 零点定理探讨了运用不变量进行射影几何定理机器证明的可能性。这种方法本质上是把 Gröbner 基的方法应用到括号代数中，这样就产生了运用拉直法则（straightening）的射影几何机器定理证明的新方法。关于拉直法则，我们将在第 2 章介绍。

1988 年 White 和 Mcmillan 运用 Cayley 分解的技巧，给出了一种几何定理机器证明的方法。1991 年，Sturmfel 和 Whiteley 把一个几何定理的结论转化为 Cayley 表达式，然后通过交、并运算转化为括号多项式，给出了射影几何定理机器证明的另外一个方法。这两个方法的缺点是证明过程中括号多项式迅速膨胀，计算和收缩都变得非常困难。

1990～1994 年，Richter-Gebert 提出了运用双二次 final 多项式的基于括号代数的射影几何定理机器证明的新方法，给出了具有几何意义的较短的可读证明。尽管这种方法不是一种完全的方法，但是适用于几乎所有的射影几何 Incidence 部分定理的证明。同时 Crapo 和 Richter-Gebert 把二次 final 多项式的方法推广到平面度量几何的定理机器证明上。他们把射影平面

中的一条固定直线作为无穷远直线，把其上的两个固定点取作原点，实现从射影几何过渡到度量几何，在理论上实现了基于括号代数的双二次 final 多项式的平面几何定理的机器证明。

2001 年，Li 在括号代数的基础上，定义了 Clifford 括号代数，并利用 Clifford 代数强大的计算功能，给出了度量几何的定理机器证明的新方法。此方法给出了五圆定理的第一个纯代数的方法。此后 Li 运用共形几何代数进行了欧氏几何定理的机器证明。这种证明更确切地可以称为推理，通过去除题设中的一个、两个甚至多个条件，然后试图通过结论将它们恢复出来。Li 和 Wu 还在[51,52]中给出了射影几何定理的有效短证明，这当中包括射影二次曲线部分的几乎所有定理。

（2）第二种方法是面积法。

面积法是 Zhang 在 1982 年就已经发现了。后来在 1992 年，Chou、Gao 和 Zhang 在此基础上给出了几何定理可读机器证明的方法，适用于射影几何、欧氏几何。1996 年，Yang、Gao、Chou 和 Zhang 把面积法推广到了非欧几何。

（3）第三种方法是 Grassmann-Cayley 代数。

1994 年，Mourrain 提出了运用 Gramer 法则的射影几何定理机器证明的方法。这种方法的特点是：①以吴方法为基础；②在消元的最后一步，引入坐标来保证方法的完全性；③没有简化消元法则，导致中间过程中项数大量膨胀[90,91]。

（4）第四种方法是 Clifford 代数。

1994 年 Li 和 Cheng 首次提出了用 Clifford 代数进行几何定理的机器证明，并且给出了一般的几何定理机器证明的框架，它综合了吴方法和 Clifford 代数。因为 Clifford 代数具有强大的

计算功能，所以从代数方面讲，这种方法包括了括号代数、面积、Grassmann-Cayley 代数三种方法。它不仅能产生可读证明，而且也适用于各种经典几何和微分几何。

1996 年，Wang、Fèvre 等利用重写法则，给出了另外一种基于 Clifford 代数的定理机器证明的新方法。1998 年，Yang、Zhang 和 Feng 给出了基于 Clifford 代数的正则化技术的几何定理机器证明的方法，它可以用于二次曲线定理的证明，但是缺点是一开始就采纳了二次曲线的一对主轴作为固定标架。

1.3 本书的内容结构与符号约定

1.3.1 内容结构

本书研究内容的出发点基于如下几点。

(1) 括号代数和仿射括号代数的除法问题，还有项数问题，一直是一个公开问题，本书想做一点这方面的理论工作。

(2) 关于射影几何和欧氏几何部分，已经有很不错的括号代数计算程序，对应于仿射几何的括号代数计算也亟须一套系统来实现。

(3) 把已经很完善的射影几何定理机器证明部分内容借鉴并推广到仿射几何中，不能推广的要建立新的理论，同时寻找更多仿射几何本身的特点和例子，发展出一套完善的计算

工具。

关于这几方面，现有的成果少之又少，特别值得一提的是吴在文献[59]中阐述了仿射括号代数以及仿射几何中表示，当然其主要内容是关于括号代数的。

本书的研究结果包括：提出了交换算法以及其他几个仿射括号代数的算法；补充和分析了大量的仿射几何的代数表示以及消去方法；讨论了括号代数以及仿射括号代数的整除性问题；给出了仿射括号代数很多新的公式和一些必要的计算理论；对仿射括号代数计算进行了程序实现。

本书所涉及问题的研究思路路线是：首先通过计算实际的例子，发现问题，通过解决这些问题得到一些好的方法、公式，通过提升，构成了本书的计算理论、公式中的一大部分。在例子中发现了现有表示的不足，促使我们去努力挖掘新的表示，构成了表示内容的绝大部分。其次借鉴括号代数的一些思想、方法，要在仿射括号代数中实现，需要进行很深入的性质研究，这部分构成了计算理论的另外一部分。最后，为了研究括号代数、仿射括号代数的整除性，我们使用了“拉直”这个基本工具，通过进一步的探索和研究，得出的结果构成了整除性理论部分。

本书的内容和结构大致如下：

第2章主要是相关基础理论的论述和探讨：2.1节主要是介绍括号代数及其理论，同时介绍了括号代数与Grassmann-Cayley代数、射影几何三者之间的关系。2.2节介绍仿射括号代数以及它的一些理论。2.3节介绍了我们在括号代数、仿射括号代数的整除性理论方面所做的工作。

第 3 章主要介绍仿射几何元素的表示、构造以及消元问题，在 Incidence 部分我们不限制维数地给出了必要的表示。在二次曲线部分，我们给出了 3 种二次曲线必要的各种表示，并且分析它们的对称性与反对称性。当然，对应于各种构造方式，我们也给出了消元方法。

第 4 章给出了仿射括号代数计算的相关理论和算法。其中 4.2 节介绍了仿射括号代数的一个关键算法——交换算法，这对于我们简化计算、统一程序实现非常有用。4.3 节讨论了必要的 σ 结构的展开问题。4.4 节讨论了仿射括号代数计算中的其他问题，包括规则形式和计算序、一些常用的公式以及多项式聚类。在 4.5 节我们集中给出了几个关键算法的具体描述：交换算法，收缩算法，GCD 提取算法，因式分解以及 3 个主算法。

第 5 章也是工作的一个主要部分。在这里讨论了程序实现等相关问题，同时还包括大约 16 个例子。5.1 节讨论了消元顺序以及我们采用的方法。5.2 节介绍了我们实现的系统以及一些说明。5.3 节介绍了程序的输入、输出，并且都给出了实际的例子。5.4 节是一些关键运算的程序代码。5.5 一节给出了例子，这些例子包括 Incidence 二维部分、Incidence 三维部分以及仿射二次曲线部分。

本书后附录部分给出了 Maple 软件介绍。

1.3.2 符号约定

贯穿始终，我们用黑体数字以及黑体大写字母表示向量，例如 [***PQR***]，[**123**] 等。

通常用**1**、**2**、**3**、**4**、**5**、**6**、**7**、**8**、**9**、**0**十个黑体数字表示向量，如果需要更多的点，可以按照ASCII码顺序加上***A***、***B***等。需要说明的是我们实现的程序也完全支持这种输入。

在第2章引入的边界算子，通常记为∂，但在本文中记为σ，原因是为了和程序保持一致。在Maple 10中，输出∂的话，会导致显示的结果部分乱码，所以我们用σ代替。

第2章 括号代数、仿射括号代数、整除性

本章概括论述了括号代数、仿射括号代数的定义以及一些相关的理论，同时分析了整除性问题，并给出了若干结果。

2.1 射影几何、括号代数与 Grassmann-Cayley 代数

群的概念出现之后，几何学逐渐被认为是研究几何图形在一些变换群下的不变的性质的学科。射影几何就是研究几何图形在射影变换群下不变的几何性质。

事实上，在几何学中，一直在考虑这个基本的问题：一个几何图形的哪些性质是射影几何的基本性质。例如在二维平面的一条直线上有 3 个点，那么这三点共线就是射影几何的性

质，因为在射影变换下这 3 点始终保持共线这个性质。从代数的观点来看这个问题：设这 3 个点是 X_1，X_2，X_3，我们需要寻找一个由此 3 个点组成的多项式，这个多项式在射影变换下不变。我们知道，其中的一个多项式就是行列式，它的几何意义就是共线。

所以推广这个观点，要寻找一个几何图形的性质，我们就可以从代数角度出发，寻找那些在变换 T 下不变的多项式。

我们先阐述一下马上要用到的符号；用 F 表示数域，没有特别说明的情况下，都是指任意特征的无限域；用 F^d 表示 F 上的 d 维向量空间；用 $SL(F^d)$ 表示 F^d 上的特殊线性群，也就是行列式为 1 的线性群，它对向量作用就是指对此向量进行左乘操作；用 $F[x_{ij}]$ 表示 F 上的多项式环。这样我们给出如下问题：

多项式环 $F[x_{ij}]$ 在 $SL(F^d)$ 作用下的不变环（记作 $F[x_{ij}]^{SL(F^d)}$）的结构如何？

此问题由不变量第一基本定理给出了解答，定理如下：

定理 2.1.1　不变环 $F[x_{ij}]^{SL(F^d)}$ 是由 $X=(x_{ij})_{d\times d}$ 的$d\times d$ 阶行列式生成的。

事实上，由这些子行列式生成的一个商代数即是括号代数。下面我们给出它的严格定义：

定理 2.1.2　构造集合如下：

$$\wedge\ (n,d)=\{[X_{i_1}\cdots X_{i_d}]:1\leqslant i_1\leqslant\cdots\leqslant i_d\leqslant n\}$$

此集合里面的元素称为括号。对于 $i_1,i_2,\cdots,i_d$ 的任意置换 π，我们有（这里 sign（π）表示置换 π 的符号）：

$$[X_{i_1}\cdots X_{i_d}] = \mathrm{sign}(\pi)[X_{\pi(i_1)}\cdots X_{\pi(i_d)}]$$

设 $F(\wedge(n,d))$ 是 $\wedge(n, d)$ 中的元素生成的域 F 上的多项式环。定义如下的代数同态：

$$\phi_{n,d}: F(\wedge(n,d)) \to F[X_{x_{ij}}]$$

$$[X_{i_1}\cdots X_{i_d}] \to \det\begin{pmatrix} X_{i_1,1} & \cdots & X_{i_d,1} \\ \vdots & & \vdots \\ X_{i_1,d} & \cdots & X_{i_d,d} \end{pmatrix}$$

$\phi_{n,d}$ 把括号映射成为 X 的 $d \times d$ 阶行列式，我们也称此为坐标化。在此映射中，$\mathrm{Ker}(\phi_{n,d}) = I_{n,d}$。它的商代数 $\frac{F[\phi_{n,d}]}{I_{n,d}}$ 被称为括号代数，通常我们记之为 $B_{n,d}$。理想 $I_{n,d}$ 被称为 $B_{n,d}$ 的合冲理想（syzygy ideal）。

定理 2.1.3 Grassmann-plücker 关系式是指如下的集合：

$$\Big\{\sum_{k=1}^{d+1}(-1)^k[X_{i_1}\cdots X_{id-1}X_{jk}]$$
$$[X_{j_1}\cdots X_{j_{k-1}}X_{j_{k+1}}\cdots X_{j_{d+1}}]:$$
$$1 \leqslant i_1\cdots \leqslant i_{d-1} \leqslant m, 1 \leqslant j_1\cdots \leqslant j_{d+1} \leqslant m\Big\}$$

不变量第二基本定理给出了以下的结论：

定理 2.1.4 不变量第二基本定理：Grassmann-plücker 关系构成了 $I_{n,d}$ 的一组基底。

不变量的两个基本定理完全解答了刚刚给出的问题，并且衍生出了括号代数的定义。由此可以看到，括号代数是研究射影几何不变量的代数工具。事实上因为它的强大计算功能，已

经被广泛应用到不同领域。下面我们列举几个它的应用以及所具有优势：

（1）括号代数可以极大地简化几何模型的表示。例如对上面的例子——三点共线的条件，如果用坐标表示，设它们的坐标分别为 $X_i = (x_i, y_i, z_i), 1 \leqslant i \leqslant 3$，那么三点共线的条件就是要满足下式：

$$- z_2x_1z_1y_3 - z_1x_2y_1z_3 + z_1^2x_2y_3 + z_2y_1z_1x_3 + y_2z_1z_3x_1 - y_2z_1^2x_3 = 0$$

但如果用括号表示，形式就简单得多，可以表示为：$[X_1X_2X_3] = 0$。另外再如空间中三线 AB，CD，EF 共点的表示，如果用坐标，将是如下48项漫长的表示：

$$\begin{aligned}
&- x_1y_2z_4y_3z_5x_6 - x_1y_2z_3x_4y_6z_5 - x_1y_2z_4x_3y_5z_6 + \\
&x_1y_2z_3y_4z_5x_6 - x_1y_2z_3y_4x_5z_6 - x_1y_3z_2x_4y_5z_6 + \\
&x_1y_3z_2x_4y_6z_5 - x_1y_3z_2z_4x_5y_6 + x_1y_3z_2z_4y_5x_6 + \\
&y_1z_2x_3y_4z_5x_6 - y_1z_2x_3y_4x_5z_6 + y_1z_2x_3z_4x_5y_6 - \\
&y_1z_2x_3z_4y_5x_6 - y_1x_2z_3x_4y_5z_6 + y_1x_2z_3x_4y_6z_5 - \\
&y_1x_2z_3y_4z_5x_6 + y_1x_2z_3y_4x_5z_6 + z_1x_2y_3x_4y_5z_6 - \\
&z_1x_2y_3x_4y_6z_5 + z_1x_2y_3z_4x_5y_6 - z_1x_2y_3z_4y_5x_6 - \\
&z_1y_2x_3y_4z_5x_6 + z_1y_2x_3y_4x_5z_6 - z_1y_2x_3z_4x_5y_6 + \\
&z_1y_2x_3z_4y_5x_6 + x_1y_4z_2x_3y_5z_6 + x_1y_2z_4x_3y_6z_5 + \\
&x_1y_2z_3x_4y_5z_6 + x_1y_2z_4y_3x_5z_6 - x_1y_4z_2x_3y_6z_5 + \\
&x_1y_4z_2z_3x_5y_6 - x_1y_4z_2z_3y_5x_6 - y_1z_2x_4y_3z_5x_6 + \\
&y_1z_2x_4y_3x_5z_6 - y_1z_2x_4z_3x_5y_6 + y_1z_2x_4z_3y_5x_6 + \\
&y_1x_2z_4x_3y_5z_6 - y_1x_2z_4x_3y_6z_5 + y_1x_2z_4y_3z_5x_6 -
\end{aligned}$$

$$y_1x_2z_4y_3x_5z_6 - z_1x_2y_4x_3y_5z_6 + z_1x_2y_4x_3y_6z_5 -$$
$$z_1x_2y_4z_3x_5y_6 + z_1x_2y_4z_3y_5x_6 + z_1y_2x_4y_3z_5x_6 -$$
$$z_1y_2x_4y_3x_5z_6 + z_1y_2x_4z_3x_5y_6 - z_1y_2x_4z_3y_5x_6 = 0$$

但是如果用括号表示将可能是：$[X_1X_2X_3][X_4X_5X_6] - [X_1X_2X_4][X_3X_5X_6] = 0$，这样的表示有三种可能，但是都简化得多。当然同时也带来如何选择表示的问题。需要特别提到的是仿射括号代数来源于括号代数，所以与括号代数一样，仿射括号代数一样具有简化表示的优势。

（2）括号代数用于抽象图不可实现的证明。我们举一个经典的例子：射影空间中 8 个点只能满足如下 5 组共面条件，这样的几何图形是否存在？

$$[X_1X_2X_5X_6] = 0, [X_1X_3X_5X_7] = 0$$
$$[X_1X_4X_5X_8] = 0, [X_2X_3X_6X_7] = 0$$
$$[X_2X_4X_6X_8] = 0$$

事实上，这样的图形是不存在的，因为在合冲理想里面存在如下的一个多项式：

$$\begin{aligned} f = & [X_1X_2X_5X_6][X_1X_3X_4X_7][X_1X_2X_4X_8][X_2X_3X_4X_7] - \\ & [X_1X_3X_5X_7][X_1X_2X_4X_6][X_1X_2X_4X_8][X_2X_3X_4X_7] + \\ & [X_1X_4X_5X_8][X_1X_2X_4X_6][X_1X_2X_3X_7][X_2X_3X_4X_7] + \\ & [X_2X_3X_6X_7][X_1X_3X_4X_7][X_1X_2X_4X_8][X_1X_2X_4X_5] - \\ & [X_2X_4X_6X_8][X_1X_3X_4X_7][X_1X_2X_4X_5][X_1X_2X_3X_7] + \\ & [X_3X_4X_7X_8][X_1X_2X_4X_6][X_1X_2X_4X_5][X_1X_2X_3X_7] \end{aligned}$$

因为 $f = 0$ 是一个恒等式，从已知条件我们马上就可以推

出这样的结论：

$$[X_3X_4X_7X_8][X_1X_2X_4X_6]$$
$$[X_1X_2X_4X_5][X_1X_2X_3X_7]=0$$

无论哪一个括号等于零，结果都表明对应的括号内四点共面，与已知条件矛盾。

（3）括号代数用于定理机器证明的研究，这方面的工作在第 1 章中给出了简单介绍。用括号代数进行定理机器证明具有表示简化、分解具有几何意义、计算简单的优势。同样这些优点仿射括号代数也具备。

射影几何的表示与括号代数之间的联系是通过 Grassmann-Cayley 代数实现的。首先我们参考文献［16，17］给出通用的 Grassmann-Cayley 代数的定义。

记 V 是域 F 上的一个 d 维向量空间，ε 是一个 C_d^k 维向量代数，$\overbrace{V\times\cdots\times V}^{k}$ 是 V 上的 k 重向量空间，其中 $k=2,3,\cdots,d$，定义 V 的 k - 级外代数是由 ε 和如下的映射 $\wedge^k$ 组成：

$$\overset{k}{\wedge}:\ \overbrace{V\times\cdots\times V}^{k}\mapsto\varepsilon$$

同时这个映射还满足如下的条件：

(1) $$\wedge^k(\boldsymbol{A}_1,\cdots,\boldsymbol{A}_i,\cdots,\boldsymbol{A}_j,\cdots,\boldsymbol{A}_k)=-\wedge^k(\boldsymbol{A}_1,\cdots,\boldsymbol{A}_j,\cdots,\boldsymbol{A}_i\cdots,\boldsymbol{A}_k)$$

$$\wedge^k(\boldsymbol{A}_1,\cdots,m_1\boldsymbol{A}_i+m_2\boldsymbol{A}_j,\cdots,\boldsymbol{A}_k)=m_1\wedge^k(\boldsymbol{A}_1,\cdots,\boldsymbol{A}_i,\cdots,\boldsymbol{A}_k)+m_2\wedge^k(\boldsymbol{A}_1,\cdots,\boldsymbol{A}_j,\cdots,\boldsymbol{A}_k)$$

其中，$(A_1,\cdots,A_i,\cdots,A_j,\cdots,A_k)\in V$；$\{i,j\}\subset\{1,\cdots,k\}$；$m_1$，$m_2\in F$ 这就是所谓的对称性和线性性。

（2）泛性质：如果φ是一个从V的k重向量空间到任意一个C_d^k维向量空间H的k重线性映射，并且具有反交换性，则存在唯一一个线性映射$g:\varepsilon\mapsto H$使得下面的图表可交换：

$$\overbrace{V\times\cdots\times V}^{k}\xrightarrow{\wedge^k}\varepsilon$$

$$\varphi\downarrow\quad\swarrow g$$

$$H$$

如果$\wedge^k$和ε满足如上的条件，那么向量空间ε根据同构则被唯一决定，被称为k－级外代数，记做$\wedge^k(V)$。任意的多重向量对于多重向量$(A_1,\cdots,A_k)$，它的像$\wedge^k(A_1,\cdots,A_k)$记作$A_1\vee\cdots\vee A_k$，称为k级外张量。

做直和如下：

$$\wedge(V)=\sum_{k=0}^{d}\wedge^k(V)$$

其中$\wedge^0(V)=F$，$\wedge^1(V)=V$。假定$x=x_0+\cdots+x_d$，$y=y_0+\cdots+y_d$是$\wedge(V)$中任意的两个元，其中$x_i,y_i\in\wedge^i(V)$，定义$\wedge(V)$中的一个乘积运算$\times$如下：

$$x\times y=\sum_{i=1}^{d}\sum_{j=1}^{d}x_i\vee y_j$$

并且$\times$满足结合律，$\times$与加法满足分配律，其乘法单位元是V的单位元，从而$\wedge(V)$成为一个代数。

记N^k是下面的集合生成的$\wedge^k(V)$的一个理想：

$\{A_1 \vee \cdots \vee A_k \subset \wedge^k(V): \exists A_i = A_j, i \neq j \in \{1, \cdots, k\}\}$

其中 $k = 0, \cdots, d$ 。

做 $\wedge(V)$ 中的理想如下：

$$N = \sum_{k=0}^{d} N^k$$

则商代数 $\frac{\wedge(V)}{N}$ 被称为外代数，或者所谓的 Grassmann 代数，其中的乘法运算 × 和每一个 k 级外代数中的 V 都是相容的，我们把这种乘法统一记作 $\vee$ ，被称为外积运算，或者并运算。这里我们把 Grassmann 代数所在的多重向量空间的并空间称为 Grassmann 空间。

我们将来用如下记号：$x \in \wedge^r(V)$ 记作 $\langle x \rangle_r$ ，I 是 $\wedge^d(V)$ 中的固定元素。二次型：

$$D(x,y) = \frac{\langle x \wedge y \rangle_n}{I}, x, y \in \wedge(V)$$

是非奇异的，可诱导出一个可逆线性映射 i：$\wedge(V) = \wedge(V^*)$ ，其中 V^* 为 V 对偶空间。那么 $\wedge = i^{-1} o \vee o\, i$ 定义了 $\wedge(V)$ 中的一个运算，称为交运算，记作 $\wedge$。

这样我们可以给出 Grassmann-Cayley 代数的定义：

定理 2.1.5　$\wedge(V)$ 及其中的元，满足并交运算组成的代数称为 Grassmann-Cayley 代数。

记 $a = A_1 \cdots A_j$ 是一个 j 级外张量，$b = B_1 \cdots B_k$ 是一个 k 级外张量，其中 $j + k \geq d$ 。则交并运算按照如下法则，可以转变为含有括号的表达式：

$$a \wedge b = \sum_{\pi} \operatorname{sign}(\pi)[\boldsymbol{A}_{|\pi(1)|} \cdots \boldsymbol{A}_{|\pi_{d-k}|} \boldsymbol{B}_1 \cdots \boldsymbol{B}_k] \times \boldsymbol{A}_{|\pi(d-k+1)|} \cdots \boldsymbol{A}_{\pi(j)} \tag{2.1}$$

其中，π 取遍 $1,\cdots,j$ 的所有置换，并且满足 $\pi(1) < \cdots < \pi(d-k) < \pi(d-k+1) < \cdots < \pi(j)$ 。这样的运算可以根据分配律推广到 $\wedge(V)$ 。

假定 $\varepsilon_1,\cdots,\varepsilon_d$ 是向量空间 V 的任意一组基底，则 $\wedge(V)$ 有下述一组基底：

$$\{E_{j_1} \vee \cdots \vee E_{j_k}: 1 \leqslant j_1 \leqslant \cdots \leqslant j_k \leqslant d\}$$

设 $\boldsymbol{A}_1,\cdots,\boldsymbol{A}_k \in \mathrm{V}$，其中每一个向量在基底下的表达式是 $\boldsymbol{A}_i = \sum_{j=1}^{d} a_{ij}E_j$，根据多重线性和反交换性，有如下的等式：

$$\boldsymbol{A}_1 \vee \cdots \vee \boldsymbol{A}_k = \sum_{1 \leqslant j_1 \leqslant \cdots \leqslant j_k \leqslant d} \begin{vmatrix} a_{1j_1} & \cdots & a_{kj_1} \\ \vdots & & \vdots \\ a_{1j_k} & \cdots & a_{kj_k} \end{vmatrix} E_{j_1} \vee \cdots \vee E_{j_k} \tag{2.2}$$

式（2.2）右侧中所有的行列式就构成了 $\boldsymbol{A}_1 \vee \cdots \vee \boldsymbol{A}_k$ 的 Grassmann-plücker 坐标；反过来，对任意一个 C_n^k 维向量，它可以是一个 k 级外张量当且仅当它的坐标满足 Grassmann-plücker 关系。

定理 2.1.6 Cayley 分解：把一个 C_n^k 维的向量转变为一个 k 级外张量，称做外张量的 Cayley 分解。把括号多项式转变为 Grassmann-Cayley 代数的表达式，则称做 Cayley 因式分解。

通常上述过程也可逆，把一个 Grassmann-Cayley 代数表达式通过运算法则可以转变为括号多项式。一个 d 级外张量 $\boldsymbol{A}_1 \vee \cdots \vee \boldsymbol{A}_d$ 可以转变为一个括号 $[\boldsymbol{A}_1 \cdots \boldsymbol{A}_d]$，对于一个低于 d 的 k 级外张量 $\boldsymbol{A}_1 \vee \cdots \vee \boldsymbol{A}_k$，有时候添加 V 中的任意 $d-k$ 个向量，构成括号 $[\boldsymbol{A}_1 \cdots \boldsymbol{A}_k \mathrm{X}_1 \cdots \mathrm{X}_{d-k}]$ 来表示，添加的向量称为哑元。

注意，本书为了方便，把 $A_1 \vee \cdots \vee A_k$ 简记为 $A_1 \cdots A_k$。

Grassmann-Cayley 代数是把一个射影几何的表达式转化为具有代数不变性质的括号形式的重要工具。Cayley 分解是它的逆过程，把一个具有不变性质的括号多项式经化为 Grassmann-Cayley 代数表示形式，进而得到对应的几何解释。所以我们认为 Grassmann-Cayley 代数是射影几何和括号代数的桥梁，在运用括号代数解决射影几何问题时不可避免地要使用 Grassmann-Cayley 代数。经典的图示如下：

射影几何↔Grassmann-Cayley 代数↔括号代数

同样仿射括号代数在运用于几何定理机器证明时也不可避免地要经过这个桥梁。

从射影几何到 Grassmann-Cayley 代数，通过直接解读就可以得到用并、交等运算（$\vee$，$\wedge$）形成的表达式。从 Grassmann-Cayley 代数到括号代数通过式（2.1）展开成括号形式。反过来，通过 Cayley 因式分解进行从括号代数到 Grassmann-Cayley 代数的转化，然后通过直接读取从 Grassmann-Cayley 代数转化为射影几何。

我们仍然用最经典的例子来说明三者之间的关系：

射影几何：三条直线 AB,CD,EF 交于一点。

↑↓

Grassmann-Cayley 代数：$(A \vee B) \wedge (C \vee D) \wedge (E \vee F) = 0$

↑↓

括号代数：$[ABE][CDF] - [ABF][CDE] = 0$ 等三个表达式

2.2 仿射括号代数及其合冲理想

把括号代数在射影几何中的运算推广到仿射几何，关键之一就是保证计算的齐次性。为此我们引入一个算子 σ，称为边界算子（Boundary Algorithm Operators，BAO）。

定义 2.2.1 假设 $E_1 \vee E_2 \vee \cdots \vee E_{n-1}$ 是域 F 上 d 维向量空间 V 的一个超平面，它由（d－1）个固定向量组成。定义 σ 是一个从 Grassmann 空间到 Grassmann 空间的映射，规则如下：

(1) σ 在每一个 Grassmann 分级空间都是线性的。

(2) $\sigma(\lambda) = 0$，如果 $\lambda \in F$。

(3) $\sigma(\lambda) = 0$，如果 $\lambda \in G^d(V)$，其中 $G^d(V)$ 指 Grassmann 第 d 个分级空间。

(4) $\sigma(A) = [AE_1E_2\cdots E_{d-1}]$，如果任意的 $A \in V$。

(5) $\sigma(A_1 \vee \cdots \vee A_k) = \sum_{i=1}^{k} (-1)^{i-1} \sigma(A_i) A_1 \cdots \check{A}_i \cdots A_k$，

如果 $A_1 \vee \cdots \vee A_k \in G^k(V)$ 。其中 $G^k(V)$ 是指第 k 个分级空间，$k = 2,\cdots,d$ 。同时用符号 $\check{A}_i$ 表示不含有 A_i 的组合。

如果 σ 满足上述所有性质，它就被称为边界算子，它所作用的量称为边界量。在此处以及以后的章节，在不引起混淆的情况下，我们把 $A_1 \vee \cdots \vee A_k$ 记为 $A_1 \cdots A_k$ 。

事实上，$E_1 \vee E_2 \vee \cdots \vee E_{n-1}$ 就是几何意义上的无限远。例如 $d = 2$，它就是直线上的无穷远点。如果 $d = 3$，它就是平面上的无穷远直线。$d = 4$，对应的就是无穷远平面。

引理 2.2.2　$\sigma^2(A_1 \cdots A_k) = 0$

证明： 证明从定义就可以得出。

在引入边界算子后，我们可以定义仿射括号代数。

定义 2.2.3　设 $X_1, X_2, \cdots, X_n$ 是域 F 上向量空间 V 中的 n 个向量。$\phi_{n,d}$，$I_{n,d}$ 如括号代数的定义。给出两个新的集合：

$$\phi_{n,d}^{e} = \sigma(X_1), \cdots, \sigma(X_n)$$

$$G^{e} = \left\{ \sum_{k=1}^{d+1} (-1)^{k+1} [X_{i_1} \cdots X_{i d+1}] \sigma(X_{i_k}) : \right.$$

$$\left. 1 \leqslant i_1 \leqslant \cdots \leqslant i_d \leqslant n \right\}$$

$\phi_{n,d}^{a} = \phi_{n,d} \cup \phi_{n,d}^{e}$ 。如果 $I_{n,d}^{e}$ 是一个由 G^{e} 生成的理想，$I_{n,d}^{a}$ 是由 $I_{n,d}$ 中的元素和 $I_{n,d}^{e}$ 的元素生成的理想。类似于括号代数的定义，我们可以定义仿射括号代数为商代数 $\dfrac{F[\phi_{n,d}^{a}]}{I_{n,d}^{a}}$ 。$I_{n,d}^{a}$ 被称为仿射括号的合冲理想。

我们知道在括号代数中，$I_{n,d}$ 有已知的如下三种形式的基

底，这里 $[\cdots\check{X}_i\cdots]$ 表示 X_i 不出现在括号中。

(1) Van der Warerden 合冲基底 $VW_{n,d}$。

$$[[XYZ]] := \sum_{\pi \in \Lambda(d+1,s)} \operatorname{sign}(\pi,\pi^*) \times [X_1\cdots X_{s-1}Y_{\pi_1^*}\cdots Y_{\pi_{d+1-s}^*}] \times [Y_{\pi_1}\cdots Y_{\pi_s}Z_1\cdots Z_{d-s}]$$

(2) 两个拉直基底。

$$S_{n,d} := \{[[XYZ]] \in VW_{n,d}: X_{s-1} < Y_{s+1}, Y_s < Z_1\}$$

$$S_{n,d}^* := \{[[XYZ]] \in S_{n,d}: X_i < Y_i, 1 \leqslant i \leqslant s-1\}$$

(3) Grassmann-plücker 关系。

$$GP_{n,d} := \left\{ \sum_{k=1}^{d+1} (-1)^k [X_{i_1},\cdots,X_{i_{d-1}}X_{j_k}] [X_{j_1}\cdots \check{X}_{j_k}\cdots X_{j_{d+1}}\right\}: 1 \leqslant i_1 \leqslant \cdots \leqslant i_{d-1} \leqslant n, 1 \leqslant j_1 \leqslant \cdots \leqslant j_{d+1} \leqslant n$$

为了说明仿射括号代数的基底，我们引入新的记号：

$$S_{n,d}^a = S_{n,d} \cup G^e$$

$$S_{n,d}^{*,a} = S_{n,d}^* \cup G^e$$

$$GP_{n,d}^a = GP_{n,d} \cup G^e$$

在仿射括号代数中，有两个 Grassmann-plücker 关系：

$$\sum_{i=1}^{d+1} (-1)^i [X_1\cdots X_{d-1}X_i][V_1\cdots \check{V}_i\cdots V_{d+1}] = 0$$

$$\sum_{i=1}^{d+1} (-1)^i [V_1\cdots \check{V}_i\cdots V_{d+1}]\sigma(V_i) = 0$$

基于此，有仿射括号代数的两个基本运算：扩张和收缩。

定义 2.2.4　扩张变换：设 $V_1,\cdots,V_d$ 是 F^d 空间中的任意 d 个向量且它们的括号 $[V_1\cdots V_d]\neq 0$。则我们可以对括号多项式 P 中任意的包含 X 的括号 $[X_1\cdots X_{d-1}X]$ 进行如下操作：

$$[V_1\cdots V_d][X_1\cdots X_{d-1}X]=\sum_{i=1}^{d}(-1)^{i+1}[V_iX_1\cdots X_{d-1}][V_1\cdots\check{V}_i\cdots V_dX] \tag{2.3}$$

$$[V_1\cdots V_d]\sigma(X)=\sum_{i=1}^{d}(-1)^{i+1}[V_1\cdots\check{V}_i\cdots V_dX]\sigma(V_i) \tag{2.4}$$

其中式（2.3）就是括号代数中的扩张变换，式（2.4）则是因为引入了 σ 而自然产生的扩张变换。

进行括号多项式各种变化和消点时，因为 syzygy 的存在，项数会迅速膨胀，这样必须引入收缩变换，将项数控制到最少。需要说明的是，直到现在仍然没有一种完全的方法能从理论上保证什么时候收缩到最短，如何收缩到最短。Li 与 Wu 在括号代数中引入的强收缩方法，在实际使用中效果强大[51,52]。

定义 2.2.5　收缩变换：如果一个括号多项式通过某种变换，项数严格减少，这种变换就称为收缩变换。

有时候项数增加，但是能够出现公因子，次数降低，也称为收缩变换。事实上一般两者存在着矛盾，要根据需要来决定。关于公因子的情况，我们给出了完全的方法，将在下一部

分论述。

这里我们给出两个在二维仿射平面的收缩的例子，它们都是由式（2.1）和式（2.2）推导而来：

$$[BV_1V_2][BV_3V_4]-[BV_1V_3][BV_2V_4]+[BV_1V_4][BV_2V_3]=0$$

$$[V_1V_2V_3]\sigma(V_4)-[V_1V_2V_4]\sigma(V_3)+[V_1V_3V_4]\sigma(V_2)-[V_2V_3V_4]\sigma(V_1)=0$$

Wu 提出了扩张和收缩两个变换，它们几何意义就是对点集满足交比或者单比的等式进行变换，并且给出了如下两个理论结果：

定理 2.2.6 收缩定理一：d, n 是两个正整数，满足条件 $2\leqslant d\leqslant n$，$I_{n,d}$ 是括号代数 $B_{n,d}$ 的合冲理想，P 是任意一个括号多项式，如果 $p\in I_{n,d}$，则 p 乘以一个括号多项式能被收缩为零。

定理 2.2.7 收缩定理二：d, n 是两个正整数，满足条件 $2\leqslant d\leqslant n$，$I^a_{n,d}$ 是括号代数 $B^a_{n,d}$ 的合冲理想，P 是任意一个括号多项式，如果 $p\in I^a_{n,d}$，当 $d\leqslant 4$ 时则 p 乘以一个括号多项式能被收缩为零。

2.3 括号代数、仿射括号代数的整除性理论

对于括号代数的运算，乘法通常情况下更有意义，因为它

可以给出需要的几何解释；而加、减法相对较麻烦，事实上，加减法要通过一系列的扩张和收缩才能给出我们需要的几何解释。那么除法能提供给我们什么信息？考虑除法是因为如下的原因：无论是在消元还是计算，我们都希望括号多项式的次数降低（另外一个是项数减少），这种降低一般是通过提取公因子，在几何定理证明时，这个公因子不等于零就自然是非退化条件。我们希望在每次运算时都能进行提取公因子的操作，这就出现了问题，因为括号代数或者仿射括号代数中因为syzygy的存在，一个多项式是否能以某一个括号 B 为公因子是不显然的。提取公因子的本质就是用括号 B 去除多项式的各个单项式，这当中可能某个单项式或者某些单项式需要通过变换才能整除。

我们的目的是要给出一个完全的方法来判断一个括号 B 是否能整除括号多项式 P，思路是通过拉直算法。首先我们考虑括号代数的整除性。

定理 2.3.1　对于 $\Lambda(n,d)$ 中的元素我们给出一个序，这就是字典序。也就是说，括号 $B_1=[\lambda],B_2=[\mu]$。如果 $[\lambda]<[\mu]$，是指存在一个 $m,1\leqslant m\leqslant d$，满足 $\lambda_j=\mu_j,1\leqslant j\leqslant m-1$，且 $\lambda_m<\mu_m$。这就指定了 $\Lambda(n,d)$ 中元素的一个总序，由此所诱导的单项式的反字典序我们仍然记作符号“$<$”。通常由此所定义的单项序也叫做tableaux order。

tableaux order可以把 $\Lambda(n,d)$ 中的单项式写成一个矩阵或者列表的形式。例如有如下括号：$[\lambda^1],\cdots,[\lambda^k]\in\Lambda(n,d)$，$[\lambda^1]\leqslant\cdots\leqslant[\lambda^k]$。则对应的一系列单项式 $T:=[\lambda^1][\lambda^2]\cdots[\lambda^k]$，其中 $\lambda^i=[\lambda_1^i\cdots\lambda_d^i],1\leqslant i\leqslant k$。可以写成如下

列表（tableaux）形式：

$$T=\begin{pmatrix}\lambda_1^1 & \cdots & \lambda_d^1\\ \lambda_1^2 & \cdots & \lambda_d^2\\ \vdots & & \vdots\\ \lambda_1^k & \cdots & \lambda_d^k\end{pmatrix}$$

一个列表（Tableaux）称为是标准的，是说它的每一列都已经排序，也就是说：$\lambda_s^1 \leqslant \lambda_s^2 \leqslant \cdots \leqslant \lambda_s^k$ 对所有的 $s=1,2,\cdots,d$ 都成立，否则就称为不标准的。不标准也称为不直，也就是需要拉直。

序的定义可以让我们给出一个可以降序的过程，拉直算法就是要把不标准或者说不直的多项式拉直，拉直的结果中所有的单项式序都比原来的序要低，这个序是指 Tableaux 序。

在 Van der Waerden 的 syzygy 中

$$\begin{aligned}[[X\dot{Y}Z]]:=&\sum_{\pi\in\Lambda(d+1,s)}\operatorname{sign}(\pi,\pi^*)\times[X_1\cdots X_{s-1}Y_{\pi_1^*}\cdots Y_{\pi_{d+1-s}^*}]\\ &\times[Y_{\pi_1}\cdots Y_{\pi_s}Z_1\cdots Z_{d-s}]\end{aligned}$$

上式如果满足 $X_{s-1}<Y_{s+1}, Y_s<Z_1$ 则称为拉直 syzygy，我们正是通过此过程进行多项式的拉直。

拉直算法

输入：一个括号多项式 P。

输出：已经拉直的括号多项式。

步骤 1：按照上述的序对 P 排序，并且确定 n 和 d。

步骤 2：按照拉直 syzygy 对 P 的每一个单项式进行拉直，结果仍然记为 P。

步骤 3：如果 P 中仍然有不直的单项式，转步骤 2，否则转步骤 4。

步骤 4：结束，输出结果。

程序最后必将终止，因为运算每一次序都严格降低。

举一个例子，$n=6, d=3, P=[145][156][234]+[124][145][356]$。我们可以看出这个多项式 P 中的第一项不是标准的，因为 $\lambda_2^2=5>\lambda_2^3=3$，而第二项是标准的，所以我们只需对第一项进行拉直，记第一项为 $T=[145][156][234]$。

$$
\begin{aligned}
[[1\dot{5}\dot{6}\dot{2}\dot{3}4]] = {} & [156][234]+[136][245]-[135][246]- \\
& [126][345]+[125][346]+[123][456]
\end{aligned}
$$

$$
\begin{aligned}
[145][156][234] = {} & [136][145][245]+[135][145][246]+ \\
& [126][145][345]-[125][145][346]- \\
& [123][145][456]
\end{aligned}
$$

但是第一项，第三项仍然不直，我们继续拉直：

$$
[136][145]=[135][146]-[134][156]
$$

$$
[126][145]=[125][146]-[124][156]
$$

最终我们可以得到拉直结果：

$$
\begin{aligned}
[145][156][234] = {} & [123][145][456]-[124][145][356]+ \\
& [134][145][256]
\end{aligned}
$$

在我们的系统中已经集成了拉直算法，可以很方便地对任意多项式进行拉直，注意以上所述都是在 $1<\cdots<\cdots$ 这样的自然序下。

引理 2.3.2　设在 $X_1<X_2<\cdots<X_n$ 的序下，括号多项式

P 可以最终拉直为 $P = M_1 + M_2 + \cdots + M_k$ 的形式，那么新的括号多项式 $Q = [X_1 \cdots X_d] \cdot P$ 最终的拉直结果必为如下形式：

$$[X_1 \cdots X_d] \cdot P = [X_1 \cdots X_d] \cdot M_1 + \cdots [X_1 \cdots X_d] \cdot M_k$$

证明：证明是简单的，首先根据拉直算法逐步降低 tableaux 序，使之降到最小，所以拉直的结果具有唯一性。

此外，因为在一个括号中，不可能出现重复的向量，否则为零，所以 $[X_1 \cdots X_d]$ 在所有的 Q 的可能括号中具有最低的序，也就是说对于每一个 Q 可能的单项式中（每一个都含有 $[X_1 \cdots X_d]$），拉直不会改变 $[X_1 \cdots X_d]$，仅会改变原来 P 中的单项式。所以结果成立。证毕。

上述引理的成立是因为拉直的唯一性和拉直的严格降序操作。有了这个引理，我们就能够给出一个括号整除一个括号多项式的充要条件。

定理 2.3.3 一个括号 $B = [X_1 \cdots X_d]$ 整除括号多项式 P，当且仅当在 $X_1 < \cdots < X_d < \cdots < X_n$ 的序下，完全拉直 P 后的多项式中的每一个单项式都包含 B。

证明：因为拉直是 syzygy 变化，所以充分性显然。

必要性证明：

因为 B 能整除 P，则 P 必然是在一些变化下具有这样的形式：$P = M_1B + \cdots + M_kB$。

如果 $M_1, \cdots, M_k$ 本身就是直的，则因为 B 是在我们所取得序下是最低的序，所以每一个单项式都含有 B，根据拉直的唯一性命题成立。

如果不直，对每一个单项式 $M_i, 1 \leqslant i \leqslant k$ 进行拉直，因为

在我们所取得序下，B 是所有括号中最低的，根据引理，拉直后的结果每一个单项式必然包含 B，再根据拉直的唯一性，命题成立。

综合两种情况必要性得证。

我们看以下几个例子：

【例 2.1】 $P := [126][345] + [124][356] - [125][346]$。

我们来判定是否括号 $B = [123]$ 能整除于它，这样就取自然序 $1 < 2 < \cdots < 6$ 就可以了，在三项里面只有第一项不直，所以只需要拉直第一项。

$$[126][345] = [125][346] + [123][456] - [124][356]$$
$$P := [123][456]$$

所以 $[123]$ 能够整除 P，同时能够看出 $[456]$ 也能整除 P。

【例 2.2】 $Q := [124][356] - [134][256] - [123][456]$。

我们来判定 $[156]$ 是否能整除 Q，取这样的序 $1 < 5 < 6 < 2 < 3 < 4$。这样可以看出 Q 重排序为：$Q = [124][563] - [134][562] - [123][564]$。每一项都不直。

$$[124][563] = [163][524] - [153][624] - [156][234]$$
$$[134][562] = [156][234] - [152][634] + [162][534]$$
$$[123][564] = [163][524] + [152][634] - [162][534] - [156][234] - [153][624]$$

最后计算 Q 的结果为：

$$Q = -[156][234]$$

也就是说 $[156]$ 能够整除 P，同时 $[234]$ 也必能整除 Q。

【例 2.3】 看下面的括号多项式:

$$\begin{aligned} P := &[345][236][125][125][346] + \\ &[345][236][125][235][146] - \\ &[135][235][245][126][346] - \\ &[125][345][123][256][346] \end{aligned}$$

它是否包含公因子 [235]?

事实上，它是包含公因子 [235] 的，取定序 $2<3<5<1<4<6$，对上述括号多项式进行拉直，拉直结果为:

$$\begin{aligned} P := &[235][235][214][316][546] - \\ &[235][231][154][356][146] \end{aligned}$$

所以可以判断出含有公因子 [235]。事实上它可以变化为:

$$\begin{aligned} P := &[235]([136][246][125][345] - \\ &[126][135][245][346]) \end{aligned}$$

这表示 6 点 1，2，3，4，5，6 共二次曲线或者 3 点 2、3、5 共面。

从上面的例子我们可以看出这样一个现象：用于判定所需要的序其实有很多种，只要满足括号 B 在相应的序下最低即可。

推论 2.3.4 上述判定定理所用的序有多种，其序可以写成如下的形式:

$$X_{\pi(1)} < \cdots < X_{\pi(d)} < X_{\tau(d+1)} < \cdots < X_{\tau(n)}$$

其中，设括号 $B=[X_1\cdots X_d]$，P 中所含有的向量为 $X_1,\cdots,X_n$，

π，τ 是任意的置换。

证明：上述推论能够成立，因为条件中的所有序都不会改变 $B = [X_1 \cdots X_d]$ 在 P 中序最低的事实。

上述关于括号的整除性的给出，事实上定义了括号代数中的除法。下面我们考虑不再是单个括号的情形：单项式之于括号多项式。事实上上述关于单个括号的整除性，如果结果为单项式，则根据齐次性，必然同时给出另外的公因子。

对于一个单项式 $M = B_1 \cdots B_k$，同时设 $B_i = [X_{i1} \cdots X_{id}]$，$1 \leqslant i \leqslant k$。我们的解决方法有两种，一种是利用上述单个括号的结果，不断重复的过程来判断，下面是对应算法。

单项式整除判定算法

输入：多项式 P，单项式 $M = B_1 \cdots B_k$。

输出：判断是否能整除，能够整除输出 true，否则输出 false。

步骤1：$i = 1$。

步骤2：如果 $i > k$，转步骤4，否则按照使 B_i 序最低的序 $X_{i1} < \cdots < X_{id} < \cdots$ 对 P 进行完全拉直。

步骤3：如果存在拉直后的结果中的某一个单项式不含有 B_i，转步骤4；如果都含有，则每一个单项式都去掉 B_i，组成一个新的多项式，仍然记为 P，$i := i + 1$，转步骤2。

步骤4：如果 $i = k + 1$，输出 true，否则输出 false。

另外一种是基于序的分析方法，针对 P 的向量构成，有以下两种可能情形。

第一种情况对于每一个 P 中的向量是多重线性的话，根据前面的思路，我们仍然可以给出一个序，使得单项式 M 在所

有 P 的可能的单项式中 tableaux 序最低。在不是多重线性的情况，我们可以给出一个序，使得 M 在 P 所有可能的单项式序中最低。

引理 2.3.5 针对括号代数多项式 P 的某一个单项式，我们总能定义 P 所包含的向量的一个序，使得在此序下，此单项式的 tableaux 序最低。

证明： 我们针对两种情况来讨论。

首先，如果对各个向量是多重线性，比如单项式具有形式：$M = B_1 \cdots B_k, B_i = [X_{i1} \cdots X_{id}], 1 \leqslant i \leqslant k$，我们定义序为 $X_{11} < \cdots < X_{1d} < X_{21} < \cdots < X_{kd}$。在此序下，显然单项式 M 具有最低的序。

如果各个向量不是多重线性，比如单项式仍然具有上述的形式，仍然这样排序：$X_{11} < \cdots < X_{1d} < X_{21} < \cdots < X_{kd}$，然后进行剔除。如果前面有重复向量，则剔除，重复下去，直到没有重复向量为止，最后得到的序就是我们需要的。我们来证明这种情况下 M 仍然是最低的。任意找一个 P 的单项式 M'，并且排列好此单项式的序，然后比较两者的第一项括号，显然在我们的序下，M 的序不会高于 M'。只有两种可能，如果 M' 的第一项括号的序高，命题得证；如果 M' 的括号序和 M 的相同，也就是第一个括号都相同，则它对比较序没有意义，我们可以不考虑，然后从第二项考虑，这样一直下去，到最后如果 M' 不等于 M，则必有 M' 的序高于 M。证毕。

这样，同样可以给出单项式整除括号多项式 P 的整除性判定准则。首先明确几个概念：

定义 2.3.6 一个向量 V 在多项式 P 中的次数 degree（V,

P）定义为它在每个单项式出现的次数（因为 P 是齐次的）。

单项式的次数定义为此单项式含有的括号个数和 σ 个数总和。

一个多项式 P 的次数是指它某一个单项式含有的括号个数和 σ 个数总和，因为是齐次的，任意取一个单项式即可。

定理 2.3.7 如果一个单项式次数低于一个括号多项式的次数，则具有 $M = B_1 \cdots B_k$，$B_i = [X_{i1} \cdots X_{id}]$，$1 \leq i \leq k$ 形式的单项式整除括号多项式 P，当且仅当在我们上述定义的序下完全拉直 P 后的多项式，每一个单项式都包含 B。同样此序并不是唯一的，针对任何一个括号内的向量都可以进行置换，同时单项式内括号的排列也会导致新的序。

证明： 证明与上面类似。

有了单项式整除判断，下面我们考虑多项式对多项式的整除性判断。

多项式之间整除性判定算法

输入：$P = M_1 + \cdots + M_k, Q = N_1 + \cdots + N_s$，$P$ 的次数高于 Q 的次数。

输出：是否能整除，给出判断。

取一个单项式，例如 N_1，按照此单项式的序对整个 P 进行拉直，如果拉直结果后任意单项式都不包含 N_1，则 Q 不能整除 P。否则，提取所有包含 N_1 的项，记剩下的多项式仍然为 P，然后对那些所有包含 N_1 的项去掉 N_1，记结果为 R_1。

依次类推，我们得到一系列 R_1，…，R_S。

如果 $R_1 = \cdots = R_S$，则 Q 能整除 P，否则不能整除。

关于判断两个多项式能否相等，类似上面的过程。事实上

上述算法是一个降低次数的过程，判断相等就把次数降到了最低。

下面我们考虑仿射括号代数的整除问题，在此设 Q 整除 P。

仿射括号代数的括号多项式可能有如下三种形式：

第一种是仅含有括号，事实上这是射影的情形。

第二种是仅含有 σ 结构。

第三种是既含有括号也含有 σ 结构。

我们分别讨论如下：

如果 P, Q 都是第一种情况，这是括号代数范畴，属于我们上面所讨论内容。

如果 P, Q 都是第二种情况，这个时候不存在 syzygy 变化，就是简单的数的整除，容易判定。

如果 P 是第一种情况，Q 是第二种情况，则不能整除；同样，如果 P 是第二种，Q 是第一种也不能整除。

如果 P 是第一种，或者第二种，Q 是第三种，也是不能整除。

如果 P 是第三种，Q 是第一种，这时候我们不用考虑 σ 结构的影响，只需要考虑每个单项式的括号部分即可。判断方法就是把按照 Q 的括号，建立一个序，然后对括号部分进行拉直，与两者都是括号类似。

如果 P 是第三种，Q 是第二种，例如：$P:=[123][456]\sigma(1)+[156][134]\sigma(2)$，$Q:=\sigma(1)$ 这种形式。我们使用如下方法解决这个问题：记 element（P）为 P 中所有向量的集合。

判定方法

首先，引入两个向量 $V_1, V_2 \notin \text{element}(P)$，$V_1, V_2 \notin \text{element}(Q)$，然后将 P 中的 σ 结构都写成括号的形式，例如 $\sigma(V)$ 写成 $[V_1V_2V]$，把结果仍然记为 P。把 Q 也写成类似的形式，例如 Q 写成 $[V_1V_2X]$。

然后按照序 $X < V_1 < V_2 < \cdots$ 对 P 进行拉直。这个序的定义可以类似于前面括号的部分。

如果拉直的结果中含有 $[V_1V_2X]$，则说明 Q 能整除 P，否则不能。

这个方法的合理性来自于 σ 结构的定义，把 V_1, V_2 看做无穷远点，因为是齐次运算，所以如果拉直后存在 $[V_1V_2X]$，则 V_1, V_2 不会被分配到其他括号中。

如果 P 是第三种，Q 是第三种，也是最复杂的一种、最常见的一种、首先需要判断 P，Q 的次数，如果 P 的次数低则不存在整除的可能性。然后可以用类似上面的方法来处理，序的选择按照如下的形式：

$$X < V_1 < V_2 < X_{11} < X_{12} < X_{13} < \cdots$$
$$< X_{i1} < X_{i2} < X_{i3} < \cdots$$

这里设 Q 具有 $[X_{11}X_{12}X_{13}]\cdots[X_{s1}X_{s2}X_{s3}]\sigma(X)$ 的形式。

这个方法可以推广到任意有限维的情形。

另外，还需要考虑这样一个问题，虽然这已经不属于多项式整除性判定的问题，但是对于计算还是非常重要的。那就是如果我们考虑的整除性问题并不是仅限于上面的形式，比如在二维仿射平面上还可能存在二阶 σ 结构的可能。那么关于这

个整除性有什么变化?

事实上对于拉直算法，可以类似地推广到仿射的情形。

定理 2.3.8 对于二维仿射平面上，多个 2 阶 σ 结构之间也可以定义 tableaux 序；对于 $d+1$ 维情形，多个 d 阶 σ 结构也可以类似定义 tableaux 情形。设 $\sigma(V_{11}\cdots V_{1d})$ 一直到 $\sigma(V_{s1}\cdots V_{sd})$，它们可以排列如下：

$$\begin{pmatrix} V_{11} & \cdots & V_{1d} \\ \vdots & & \vdots \\ V_{s1} & \cdots & V_{sd} \end{pmatrix}$$

如果对于上述排列的矩阵，每一列都是排好序的，那么就称它们是标准的，否则就是不标准的。

定理 2.3.9 对于括号和 σ 混合的情况，记 $B=[Y_{11}\cdots Y_{1(d+1)}]\cdots[Y_{k1}\cdots Y_{k(d+1)}]$ 为单项式的 M 部分。σ 部分是 $S=\sigma(V_{11}\cdots V_{1d})\cdots\sigma(V_{s1}\cdots V_{sd})$，我们选取 s 个向量 $T_1,\cdots,T_s$ 并且定义它们的序比所有的向量的序都高。然后列成如下表的形式：

$$\begin{pmatrix} Y_{11} & \cdots & Y_{1(d+1)} \\ \vdots & & \vdots \\ Y_{k1} & \cdots & Y_{k(d+1)} \\ V_{11} & \cdots & T_1 \\ \vdots & & \vdots \\ V_{s1} & \cdots & T_s \end{pmatrix}$$

如果对于上述列表，每一列都是排好序的，则称是标准的，否则就是不标准的。

类似的拉直算法可以应用于上面，这就可以处理需要的情形。

括号多项式本身是一种无坐标的不变量组成的多项式，考虑它的多项式化，就是想把括号多项式转化成相应的熟悉的多项式情形，并就此提出一些对应的通常的多项式问题。

括号多项式的多项式化有两种方法。一种是代入坐标，根据前面的括号代数和仿射括号代数的定义，这是很容易完成的。但是通过前面给出的例子也能看出，通常情况下，这将会导致得到的多项式非常复杂，并且如何由这个多项式返回到括号不变量的层次是一个没有解决的问题。

另外一种是把括号，或者 σ 整体看做一个变量，然后进行多项式化。通常情况下，得到的是一个带有多项式组约束的多项式。研究原来的括号多项式的性质也就变成了在一些多项式约束下的多项式的性质，这些多项式约束就是 syzygy 所生成的。

例如考虑一个多项式项数的最简单问题是在一维射影几何里面，仅有一个最简单的 syzygy：$[12][34]-[13][24]+[14][23]$，对于一个二维的括号多项式 P，所有的括号元素只能是 $[12]$、$[23]$、$[13]$、$[24]$、$[14]$、$[34]$ 6 个元素。这样我们分别设为 x_1,x_2,x_3,x_4,x_5,x_6，这样对于任意一个括号多项式 P，比如我们考虑它是否等于零，可以转化成如下的形式：

$$\begin{cases}x_1x_6-x_3x_4+x_2x_5=0\\P(x_1,x_2,x_3,x_4,x_5,x_6)=0\end{cases}$$

通常情况下，通过上述的多项式化，也就是括号代数的 syzygies 被转化成一系列多项式限制，一般这些多项式对任何一个变元都是一次的。然后我们要讨论的括号多项式 P 也相应转化成一个多项式。根据讨论的问题不同，可以分为如下几类（这里设由 syzygies 转化的多项式限制写成 $f_i(x_1,\cdots,x_n)=0$，$i=1,\cdots,k$，要讨论的多项式 P 记为 $P(x_1,\cdots,x_n)=0$）：

（1）一般情况下，我们需要考虑 P 是不是等于零，这就相当于考虑 P 是不是在 $f_1,f_2,\cdots,f_n$ 生成的理想里面。这个问题可以通过 gröbner 基等方法来解决。

（2）有时还要考虑 P 能不能分解，这相当于考虑在 f_1，$f_2,\cdots,f_n$ 的限制下多项式 P 的分解问题。

（3）对于 P 的最小项数问题：通常一个括号多项式因为 syzygy 的存在，具有不同的形式，但是可以考虑这样的情况，如果 P 的次数确定，对于每一个单项式，它包含的括号是有限的，把所有括号中的元素排列起来作为一个整体，这样 P 的所有单项式一定是所有排列的一个子集，也就是说肯定是有限项的。既然是有限项，那么一定存在一个多项式，它具有最小的项数。关于括号代数的项数问题已经存在了 100 多年，一直没有什么进展。这对于多项式来说，等于在 $f_1,\cdots,f_n$ 的限制下，求多项式 P 的最小项数的形式。

（4）类似于上面，还可以考虑 P 的最大公因子，对于多项式系统来说，就是在 $f_1,\cdots,f_n$ 的限制下，求 $P(x_1,\cdots,x_n)$ 的最大公因子。

类似的问题可以提出很多，相应的一些多项式方面的理论研究可以参见参考文献[92~95]。

第3章 表示、构造和消元

本章重点研究了几何中的构造方式、各种元素的表示以及消元问题。仿射几何分为 incidence 部分和 conic 部分。在 incidence 里面，射影几何里面交于无穷远点的两条直线变成了平行的直线，用引入的边界算子可以很好地表示这些平行关系。在 conic 部分，抛物线、椭圆、双曲线的构造方式多种多样，本章都给出了对应的表示，同时分析了表示的对称性。

3.1 引　言

本节的最终目标是用仿射括号代数进行几何计算。计算的前提需要给出各种元素的代数表示。本节给出的表示包括仿射 incidence 几何和仿射 conic 几何。

仿射几何中多了平行的概念，在射影几何中对应着两条直线交于无穷远点，或者两个平面交于无穷远直线。我们引入的

边界算子，正是为了解决这个问题。例如两条直线 AB，CD 平行，就可以方便地写为 $[AB\ \sigma(CD)]=0$，或者 $[CD\ \sigma(AB)]=0$，这就表示，CD 上的无穷远点在 AB 上，或者 AB 的无穷远点在 CD 上，也就是 $AB/\!/CD$，或者 $CD/\!/AB$，这两种表示，只相差一个符号。

本章的结构大致如下：

3.2 节给出 incidence 仿射几何的各种构造的代数表示，这其中很多是没有限定维数，但是我们针对二维平面特别给出了描述。在表示之后，给出了各种构造方式，并针对每种构造方式给出了消元规则。

3.3 节给出二次曲线的表示，这当中包括抛物线、椭圆、双曲线，并且讨论了它们的对称性。

3.4 节给出了二次曲线的各种构造方式和消元方法。

3.2 Incidence 部分的表示、构造与消元

下面列出各种 incidence 的表示如下：

(1) 在 (d－1) 维仿射空间，d 个点在同一个超平面上：

$$[1\cdots d]=0$$

$$\sigma(1\cdots d)=0$$

例如在三维仿射空间，三点 ABC 共面，就可以表示为 $[ABC]=0$。根据 σ 的定义，此时 $\sigma(ABC)=0$。

(2) 在任意有限维仿射空间中，3 是 1 和 2 的中点的表示：

$$3 \simeq \sigma(1)2 + \sigma(2)1$$

这是根据射影几何中交比的表示推出来的。

(3) 在仿射空间中，1、2、3 共线，并且 3 分 1 和 2 的单比为 r：

$$\sigma(2)[13] - r\sigma(1)[23] = 0$$

这也是利用射影几何中交比的表示，而单比定义为第 4 点是无穷远点的交比，取第 4 点为 $\sigma(12)$ 即可。

(4) 在仿射空间中，三条线 12，34，56 共点（这类似于射影情况），$12 \wedge 34 \wedge 56 = 0$，它可以有三种展开：

$$[134][256] - [234][156] = 0$$
$$[356][124] - [456][123] = 0$$
$$[125][346] - [126][345] = 0$$

(5) 在 d 维仿射空间中，A_1 是一个 $(d-1)$ 维子空间，A_2 是一个一维子空间，$A_1 /\!/ A_2$：

$$[A_1 \sigma(A_2)] = 0$$

例如 $d=2$，两条直线 12，34 互相平行就可以表示为：

$$[12\sigma(34)] = [123]\sigma(4) - [124]\sigma(3) = 0$$

如果 $d=3$，一个平面 123 与直线 45 平行可以表示为：

$$[123\sigma(45)] = [1235]\sigma(4) - [1234]\sigma(5) = 0$$

(6) 在 d 维仿射空间中，$A = A_1 \cdots A_m$ 是一个 m 维子空间，$B = B_1 \cdots B_n$ 是一个 n 维子空间，这里 $m,n < d$，$A /\!/ B$：

$$[A_1 \cdots A_m \sigma (B_i B_j)] = 0, i \neq j, 1 \leqslant i,j \leqslant n$$

或 $[B_1 \cdots B_n \sigma(A_i A_j)] = 0, i \neq j, 1 \leqslant i,j \leqslant m$。

例如 $d=3$，两个平面 123、456 平行就可以表示为：

$$[123 \sigma (45)] = 0$$
$$[123 \sigma (56)] = 0$$
$$[123 \sigma (46)] = 0$$

(7) 在仿射空间中，3 个不共线的点的重心可以表示为：

$$\sigma(2)\sigma(3)1 + \sigma(1)\sigma(3)2 + \sigma(1)\sigma(2)3$$

(8) 在 d 维仿射空间，一个过 A 点的 $(k-1)$ 维子空间平行于一个固定的 $k-1$ 维子空间 $B_1 \cdots B_k$ $(1 < k < d)$ 可以表示为：

$$A \sigma(B_1 \cdots B_k)$$

如果 $d=2$，通过 1 并且平行于固定直线 23 可以表示为：

$$1\sigma(23) = 13\sigma(2) - 12\sigma(3)$$

(9) 在仿射空间中，1，2，3，4 形成一个平行四边形，12 // 34，23 // 14，可以表示为：

$$4\sigma(2)\sigma(3)1 - \sigma(1)\sigma(3)2 + \sigma(1)\sigma(2)3$$

以下是常用的 incidence 构造方式

[CI-1 (free point)]：X 是一个自由点。

[CI-2 (semi-free point in line)]：X 在一条直线上。

[CI-3 (semi-free point in parallel line)]：X 在过点 C 并且平行于直线 AB 的直线上。

[CI-4-1 (proportion)]：X 在直线 AB 上或者在平行于 AB 的直线上，并且满足：$[XA]/[XB]=r$。

[CI-4-2 (proportion)]：X 在直线 AB 上或者在平行于 AB 的直线上，并且满足：$[XA]/[AB]=r$。

[CI-4-3 (proportion)]：X 在直线 AB 上或者在平行于 AB 的直线上，并且满足：$[XA]/[CD]=r$。

[CI-5-1 (intersection)]：X 是 AB 和 CD 的交点。

[CI-5-2 (intersection)]：X 是平行于 DE 并且通过 A 的直线与 BC 的交点。

[CI-5-3 (intersection)]：X 是经过 A 且平行 BC 的直线与经过 D 平行于 EF 的直线的交点。

下面给出对应的消元规则，这些规则适用于二维仿射平面。

消元规则1：消自由点 假如，括号多项式中含有的未定元的个数超过3个，即用如下方法进行多余的自由点的消去。首先选定3个点作为基本坐标，通常情况下，选取的是 P 中出现最多的3个点。然后按照此3点用扩张公式进行扩张，这个扩张包括括号也包括 σ 的部分。事实上，这是一个坐标化的过程，把其他的自由点用3个自由点来表示，坐标化将消除 syzygy 的影响。

消元规则2：直线上的半自由点 设 X 为直线 AB 上的半自由点，对 X 的消去有两种方法：一种是简单地用 $[ABX]=0$

展开，其实是把 X 写成 r_1A+r_2B 的形式。另外一种是参照上面坐标化的方法，在 P 中选取一个点，一般也选取出现次数最多的向量，例如 C，然后按照 ABC 进行扩张。

消元规则 3：平行直线上的半自由点 根据构造，X 在过点 C 并且平行于 AB 的直线上，对于括号多项式 P 中的任意一个括号 $[EFX]$ 用如下的消元规则。对 $[C\sigma(AB)X]=0$ 展开得到：$[C\sigma(AB)]X=[CX]\sigma(AB)-[\sigma(AB)X]C$，添加 A 后变成：$[ABC]\sigma(A)X=[ABX]\sigma(A)C-[ACX]\sigma(AB)$，代入后得到：

$$
\begin{aligned}
[ABC][EFX]\sigma(A) = {} & [ABX][CEF]\sigma(A) - \\
& [ACX]\sigma(AB\wedge EF)
\end{aligned}
$$

其中 $\sigma(AB\wedge EF)$ 的消元使用后面的消元规则。对于包含 X 的 σ 结构，用如下的消元规则：

$$[ABC]\sigma(X)=[ABX]\sigma(C)$$

消元规则 4：比例 1 X 在直线 AB 上或者在平行于 AB 的直线上，并且满足 $[XA]/[AB]=r$。对于包含 X 的任意括号 $[EFX]$，用如下的消元规则：

$$[EFX]=r[AEF]\sigma(B)+[BEF]\sigma(A)$$

对于任意包含 X 的 σ 结构：

$$(r+1)\sigma(X)=r\sigma(B)+\sigma(A)$$

消元规则 5：比例 2 X 在直线 AB 上或者在平行于 AB 的直线上，并且满足 $[XA]/[AB]=r$。对于包含 X 的任意括号 [

EFX]，用如下的消元规则：

$$[EFX] = r[AEF]\sigma(B) + (1 - r)[BEF]\sigma(A)$$

对于任意包含 X 的 σ 结构：

$$(r + 1)\sigma(X) = r\sigma(B) + \sigma(A)$$

消元规则6：比例3 X 在直线 AB 上或者在平行于 AB 的直线上，并且满足 $[XA]/[CD] = r$。对于括号多项式 P 中任意的包含 X 的括号 $[EFX]$，用如下规则进行消元：

$$[EFX] = [AEF]\sigma(D)\sigma(C) + r[CEF]\sigma(A)\sigma(D) - r[DEF]\sigma(C)\sigma(A)$$

对于含于 σ 结构中的 X，用如下方法进行消元：

$$\sigma(X) = \sigma(A) + r\sigma(C) - r\sigma(D)$$

消元规则7：交点1 X 是 AB 和 CD 的交点，$X = AB \wedge CD$。对于括号多项式 P 中任意的包含 X 的括号 $[EFX]$，用如下规则进行消元：

(1) 如果 $[EFA]$、$[EFB]$ 中有一个为零，则使用如下展开：

$$X = AB \wedge CD = [CDB]A - [CDA]B$$

例如 $[EFA]=0$，则 $[EFX] = -[CDA][EFB]$。

(2) 如果 $[EFC]$、$[EFD]$ 中有一个为零，则使用如下的展开：

$$X = AB \wedge CD = [ABC]D - [ABD]C$$

例如 $[EFC]=0$，则 $[EFX]=-[ABC][EFD]$。

（3）如果 $[ABE]$、$[ABF]$ 中有一个为零，则可以得到如下等式：

$$[EFX]=[ABE][CDF]-[ABF][CDE]$$

（4）如果 $[CDE]$、$[CDF]$ 中有一个为零，则可以得到如下等式：

$$[EFX]=[CDE][ABF]-[CDF][ABE]$$

（5）如果上述都不成立，就要考虑它们的构造顺序，例如A、B、C、D、E、F的构造顺序。因为是纯粹的计算过程，没有办法考虑几何在其中的影响，所以要看前面的构造结构。如果在以前的构造中，某对点出现的次数之和最多，比如A、B两点最多，E、F两点最少，可用如下的展开：

$$[EFX]=[ABE][CDF]-[ABF][CDE]$$

（6）σ 结构的展开，如果 $[ABC]$、$[ABD]$ 中有一个为零，不妨设 $[ABC]=0$，则用如下展开：

$$\sigma(X)=-[ABD]\sigma(C)$$

如果 $[CDA]$、$[CDB]$ 中一个为零，不妨设 $[CDA]=0$，则用如下展开：

$$\sigma(X)=-[CDB]\sigma(A)$$

否则按照 A、B、C、D 在多项式中出现的次数来决定。不妨设 A、B 在括号里面出现的次数最多，则

$$\sigma(X) = [ABC]\sigma(D) - [ABD]\sigma(C)$$

如果 A、B 在 σ 结构里面出现得多，则用：

$$\sigma(X) = [CDA]\sigma(B) - [CDB]\sigma(A)$$

消元规则 8：交点 2　在此构造中，$X = BC \wedge A\sigma(DE)$，它可以展开成为：$[ABC]\sigma(DE) - [BC\sigma(DE)]A$。当然它也可以展开成更简单的形式，但是有时候会带来麻烦。

对于每一个包含 X 的括号，例如 $[EFX]$，我们用下列规则消去点 X：

如果 $[PQA]=0$，用：

$$[PQX] = [PQ\sigma(DE)][ABC]$$

如果 $[PQB]=0$，用：

$$[PQX] = -[PQC][AB\sigma(DE)]$$

如果 $[PQC]=0$，用：

$$[PQX] = [PQB][AC\sigma(DE)]$$

如果 $[PQD]$，$[PQE]$ 中有一个为零，例如 $[PQD]=0$，用：

$$[PQX] = -[PQA][BC\sigma(DE)] - [PQE][ABC]\sigma(D)$$

否则，我们考虑 5 个点在括号还是在 σ 中出现的次数来决定，类似于上面过程。对于 $\sigma(X)$ 用下列展开：

$$\sigma(X) = [BCD]\sigma(E)\sigma(A) - [BCE]\sigma(D)\sigma(A)$$

消元规则9：交点3 此时 $X = A\sigma(BC) \wedge D\sigma(EF)$，这时可以把它看做两条直线的交点来利用交点1消元规则。注意此时把 σ 结构整体看做一个点，然后考虑是否有含有 σ 的括号有单项展开，或者是否有相对较少项展开即可（后续章节将有论述）。

3.3 仿射二次曲线的仿射括号代数表示

仿射几何除了 incidence 部分外，还有二次曲线部分和二次曲面部分，这里仅仅考虑二次曲线。关于二次曲面，无论在射影几何还是仿射几何中，都是一个待开拓的领域。以下所有表示的来源都来自于射影几何中经典的表示，然后通过计算得出来的。首先列举射影几何 conic 部分的一些关键的表示。

在射影平面上，根据 Pascal 定理，6点1、2、3、4、5、6共二次曲线当且仅当 $12 \wedge 56$、$13 \wedge 45$、$24 \wedge 36$ 三点共线。这样通过展开 $[(12 \wedge 56)(13 \wedge 45)(24 \wedge 36)] = 0$，可以得到：

$$\begin{aligned}\text{conic}(123456) := &[135][245][126][346]\\ &- [125][345][136][246] = 0\end{aligned}$$

这样由5点1、2、3、4、5决定的二次曲线上任意一点 X 都满

足下述方程（任意三点不共线），此方程也就是射影平面上二次曲线的通用表示：

$$\mathrm{conic}(12345,X):=[135][245][12X][34X]-[125][345][13X][24X]=0 \tag{3.1}$$

以下我们都用 5 个点连写来表示由这 5 点决定的射影二次曲线。则两个点 A，B 关于二次曲线 12345 共轭当且仅当：

$$\begin{aligned}&\mathrm{conjugate}(12345,A,B)\\&=[135][245]([12A][34B]+[12B][34A])-\\&\quad[125][345]([13A][24B]+[13B][24A])=0\end{aligned} \tag{3.2}$$

射影二次曲线 12345 上过 5 点的切线可以表示为（5 不是二重点）：

$$\mathrm{tangent}(12345,5):=([134][235][245]15-[135][145][234]25 \tag{3.3}$$

如果直线 12 不与射影二次曲线 12345 相切，则 12 关于 12345 的极点可以表示为

$$\begin{aligned}\mathrm{pole}(345,12):=&[145][234][235]1+\\&[134][135][245]2-\\&[124][125][345]3\end{aligned} \tag{3.4}$$

其中式（3.1）关于 1、2、3、4、5、X 反对称；式（3.2）关于 1、2、3、4、5 反对称；式（3.3）关于 1、2、3、

4 反对称；式（3.4）关于 1、2 对称，关于 3、4、5 反对称。

仿射二次曲线有三种，它们是根据与无穷远直线的关系进行分类的。

（1）与无穷远直线相切的二次曲线，称为抛物线，过此切点的非无穷远直线称为直径。

（2）如果二次曲线与无穷远直线有两个实交点，称此为双曲线，过此两个实交点的两条直线，称为渐近线。无穷远直线的极点，也就是这两条渐近线的交点称为双曲线的中心，过中心的直线称为直径。

（3）如果二次曲线与无穷远直线有两个虚交点，称为椭圆。同样定义它的中心为无穷远直线的极点，直径为过中心的直线，直径与无穷远直线的交点的极线，如果不是无穷远直线，则被称为此直径的共轭直径。特别地在实仿射几何里面，椭圆没有渐近线。

椭圆和双曲线都称为有心二次曲线，抛物线称为无心二次曲线，没有渐近线与中心的概念，因为它们都是无穷的。

在实现的程序里面，所有的相关表示都可以自动给出，这是因为无论括号还是 σ 结构中的元素都是对点的位置的索引，所以当需要不同表示时，只需要调整这些点的位置。

说明：以下的 3 个小节给出了大量的表示，但是去掉了表示的公式编号，这可以使整体美观，并且在后面需要时可以用表示的名称来索引。同时也给出了表示的对称性，这些对称性是很容易验证的。

3.3.1 相关抛物线的仿射括号代数表示

下面给出抛物线的表示，它们都是在抛物线定理证明和计算中常用的。注意，用来表示无穷远点的 T 或者 T_1、T_2 都可以用类似 σ (AB) 的边界结构来代替。

(1) 在仿射平面上，X 在被点 1，2，3 和直径方向 T 决定的抛物线上：

$$\begin{aligned}\text{parabola}(123T,X):=&[13X][23T][2XT]\sigma(1)-\\&[13T][23X][1XT]\sigma(2)\\=&0\end{aligned}$$

以上的表示关于 1，2 是反对称的。

如果方向是由 $\sigma(45)$ 给出：

$$\begin{aligned}\text{parabola}(123(45),X):=&[13X][23\sigma(45)][2X\sigma\ (45)]\sigma(1)-\\&[23X][13\sigma(45)][1X\sigma(45)]\sigma(2)\\=&0\end{aligned}$$

(2) 在仿射平面上，经过点 1 的切线可以表示为

$$\begin{aligned}\text{paratangent}(123T,1):=&+[13T][13T]\sigma(2)12-\\&[12T][12T]\sigma(3)13\end{aligned}$$

以上的表达关于 2，3 是反对称的。

如果方向是由 $\sigma(45)$ 给出：

$$\begin{aligned}\text{paratangent}(123(45),1):=&+[13\sigma45)][13\sigma(45)]\sigma(2)12-\\&[12\sigma(45)][12\sigma(45)]\sigma(3)13\end{aligned}$$

（3）在仿射平面上，一条线 TB 交上述形式表示的抛物线于一点，这里 B 不在抛物线上，则交点可以表示成如下形式：

$$X_{TB,123T} = \sigma(2)[13T][3TB]12 \wedge TB - \sigma(3)[12T][2TB]13 \wedge TB$$

表达式关于 2，3 是反对称的。

如果方向由 $\sigma(45)$ 给出：

$$X_{\sigma(45)B,123\sigma(45)} = \sigma(2)[13\sigma(45)][3\sigma(45)B]12 \wedge \sigma(45)B - \sigma(3)[12\sigma(45)][2\sigma(45)B]13 \wedge \sigma(45)B$$

（4）在仿射平面上，一条直线 $1B$ 交上述抛物线于两点，B 不在抛物线上，则除点 1 外另外一个交点可以表示为

$$X_{1B,123T} = \sigma(2)[13T][12B]T3 \wedge 1B - \sigma(3)[12T][13B]T2 \wedge 1B$$

表达式关于 2，3 是反对称的。

如果方向由 $\sigma(45)$ 给出：

$$X_{1B,123\sigma(45)} = \sigma(2)[13\sigma(45)][12B]\sigma(45)3 \wedge 1B - \sigma(3)[12\sigma(45)][13B]\sigma(45)2 \wedge 1B$$

（5）在仿射平面上，1、2、3 不共线，过点 1 切线与过点 2 切线的交点可以表示为

$$\text{intertangent}(123T,12) := [23T]^2\sigma(1)1 - [13T]^2\sigma(2)2 - [12T]^2\sigma(3)3$$

表达式关于 1、2、3 是对称的。

（6）在仿射平面上，1、2、3 不共线，过点 1 切线与弦 23

的交点可以表示为

$$\text{tangent Cord}(123T,1,23):=-[13T]^2\sigma(2)2+[12T]^2\sigma(3)3$$

表达式关于 2、3 是反对称的。

关于抛物线表示的证明以及后面的椭圆、双曲线表示的证明都类似，这里仅仅列出抛物线表达式的推导过程如下：

在射影方式下，由 5 点 1，2，3，T,X 决定的二次曲线，由式（3.1）到式（3.4），它过 T 点的切线可以表示称为

$$\text{tangent}_{123TX,T}:=[13X][23T][2TX]1T-[23X][13T][1TX]2T$$

根据抛物线的定义，这应该是一条无限远直线，对于任意的无限远点 E：

$$[13X][23T][2TX][1TE]-[23X][13T][1TX][2TE]=0$$

这样根据边界量的定义，能够将上述表达式写成如下形式：

$$[13X][23T][2TX]\sigma(1)-[23X][13T][1TX]\sigma(2)=0$$

这就是仿射几何中抛物线的表示。上述（2）（3）（4）的表示可以类似证明，（5）（6）的表示通过直接计算获得。

仅仅依靠上述的表示，在实际计算中仍然有很多局限。下面将分析上述表示与射影情况 conic 的一些关系，有了它们就可以应用射影中的一些变换结论。其中 T 特指直径方

向，它完全可以用 $\sigma(AB)$ 表示，这里 A、B 是仿射平面的两个点。

命题 3.3.1 任意三点不共线，则 4、5 两点在由 1、2、3、T 决定的抛物线上，当且仅当 conic（12345 T）= 0 成立。

证明：充分性是显然的。下面我们证明必要性。

根据命题，4、5 在 parabola（123 T）上，则有如下两式成立：

$$\text{parabola}(123T,4) := [134][23T][24T]\sigma(1) - [13T][234][14T]\sigma(2) = 0$$

$$\text{parabola}(123T,5) := [135][23T][25T]\sigma(1) - [13T][235][15T]\sigma(2) = 0$$

将上述两式写成比例形式，把关于 4 或者 5 的括号移到等式一边，就可以得出结论。

命题 3.3.2 任意三点不共线，4、5 两点在由 1、2、3、T 决定的抛物线上，则有下列两式成立：

$$[135][2T5]\text{parabola}(123T,4) - [134][2T4]\text{parabola}(123T,5) = [13T]\sigma(2)\text{conic}(T12345)$$

$$[235][1T5]\text{parabola}(123T,4) - [234][1T4]\text{parabola}(123T,5) = [23T]\sigma(1)\text{conic}(T12345)$$

证明：直接写出 parabola（123T，4）和 parabola（123T，5）的表达式，按照命题计算即可。

事实上关于抛物线的各种定理，进行机器证明是容易的，这是因为它的表示相对简单。从例子来看，困难程度是椭圆 > 双曲线 > 抛物线。

3.3.2　相关椭圆的仿射括号代数表示

下面给出椭圆的表示，它们是在以后计算中常用的，但是表达要比抛物线复杂得多。

（1）在仿射平面上，经过 5 点 1，2，3，4，5 的椭圆可以表达为

$$\text{ellipse}(12345,X):=+[125][345][13X][24X]-[135][245][12X][34X]=0$$

它关于 1，2，3，4，5 五点反对称。

它的中心可以表示为

$$\begin{aligned}O(12345)=&h_1(-h_1\sigma(1)+h_2\sigma(2)+h_3\sigma(3))1+\\&h_2(h_1\sigma(1)-h_2\sigma(2)+h_3\sigma(3))2+\\&h_3(h_1\sigma(1)+h_2\sigma(2)-h_3\sigma(3))3\end{aligned}$$

上述表示关于 1，2，3 是对称的。

它通过点 1 的直径可以表示为

$$\begin{aligned}\text{diameter}(12345,1)=&h_2(h_1\sigma(1)-h_2\sigma(2)+h_3\sigma(3))12+\\&h_3(h_1\sigma(1)+h_2\sigma(2)-h_3\sigma(3))13\end{aligned}$$

此处 $h_1=[145][234][235]$，$h_2=[134][135][245]$，$h_3=[124][125][345]$，上述表达关于 2，3 是对称的。

它经过其上点 1 的切线可以表示为

$$\text{ellitangent}(12345,1) := +[134][135][245]12 - [124][125][345]13$$

这个表示关于 2，3 是反对称的。

经过它上面两点 1 和 2 的切线的交点可以表示为（如果不平行）

$$\text{intertangent}(12345,12) := [145][234][235]1 + [134][135][245]2 - [124][125][345]3$$

这个表示关于 1，2 是对称的，关于 3，4，5 是反对称的。

（2）在仿射平面上，经过不共线 1，2，3 三点，具有中心 O 的椭圆可以表示为

$$\begin{aligned}O(12345) := &+2[123][12O][1OX][2OX]\sigma(3) - \\ &[123][123][1OX][2OX]\sigma(O) - \\ &[12X][12X][13O][23O]\sigma(O) + \\ &2[12X][1OX][13O][23O]\sigma(2) - \\ &2[12X][2OX][13O][23O]\sigma(1)\end{aligned}$$

上述表示关于 1，2，3 是对称的。

它经过点 1 的直径是可以表示为

$$\text{diameter}(123O,1) := 1O$$

$1O$ 的共轭直径可以表示为

$$\text{con-diameter}(123O,1) := +[13O][13O]\sigma(2)\sigma(2)1O -$$

$$[12O][12O]\sigma(3)\sigma(3)1O-$$
$$[13O][23O]\sigma(1)\sigma(1)2O-$$
$$[12O][23O]\sigma(1)\sigma(1)3O$$

它关于 2，3 是反对称的。

它经过点 1 的切线可以表示为

$$\text{ellitangent}(123O,1):=+2[13O][13O]\sigma(2)12+[123][13O]\sigma(O)12-2[12O][12O]\sigma(3)13+[123][12O]\sigma(O)13$$

它关于 2，3 是反对称的。

3.3.3　相关双曲线的仿射括号代数表示

双曲线的表达要比椭圆容易，比抛物线复杂，它的表示也是最多的。我们将分别给出。

(1) 在仿射平面上，经过 5 点 1，2，3，4，5 的双曲线可以表示为

$$\text{hyperbola}(12345,X):=+[125][345][13X][24X]-[135][245][12X][34X]$$

上述表示关于 1，2，3，4，5 是反对称的。

它的中心可以表示为：

$$O(12345)=h_1(-h_1\sigma(1)+h_2\sigma(2)+h_3\sigma(3))1+h_2(h_1\sigma(1)-h_2\sigma(2)+h_3\sigma(3))2+$$

$$h_3(h_1\sigma(1)+h_2\sigma(2)-h_3\sigma(3))3$$

上述表示关于 1，2，3 是对称的。

它经过点 1 的直径可以表示为

$$\text{diameter}(12345,1)=h_2(h_1\sigma(1)-h_2\sigma(2)+h_3\sigma(3))12+h_3(h_1\sigma(1)+h_2\sigma(2)-h_3\sigma(3))13$$

此处 $h_1=[145][234][235]$，$h_2=[134][135][245]$，$h_3=[124][125][345]$。

它经过点 1 的切线可以表示为

$$\text{hypertangent}(12345,1):=+[134][135][245]12-[124][125][345]13$$

上述表示关于 2，3 是反对称的。

（2）在仿射平面上，由不共线的 3 点 1，2，3 以及中心 O 决定的双曲线可以表示为

$$\begin{aligned}\text{hyperbola}(123O,X):=&+2[123][12O][1OX][2OX]\sigma(3)-\\&[123][123][1OX][2OX]\sigma(O)-\\&[12X][12X][13O][23O]\sigma(O)+\\&2[12X][1OX][13O][23O]\sigma(2)-\\&2[12X][2OX][13O][23O]\sigma(1)\end{aligned}$$

上述表示关于 1，2，3 是对称的。

它经过点 1 的直径是 $1O$。

$1O$ 的共轭直径可以表示为

$$\text{con-diameter}(123O,1):=+[13O][13O]\sigma(2)\sigma(2)1O-[12O][12O]\sigma(3)\sigma(3)1O-[13O][23O]\sigma(1)\sigma(1)2O-[12O][23O]\sigma(1)\sigma(1)3O$$

上述表示关于 2，3 是反对称的。

它经过点 1 的切点可以表示为

$$\text{hypertangent}(123O,1):=+2[13O][13O]\sigma(2)12+[123][13O]\sigma(O)12-2[12O][12O]\sigma(3)13+[123][12O]\sigma(O)13$$

上述表达关于 2，3 是反对称的。

（3）在仿射平面上，经过点 1 并且具有两条渐近线 23，45 的双曲线可以表示为

$$\text{hyperbola}(1,23,45,X):=[123][145]\sigma(X)^2-[23X][45X]\sigma(1)^2=0$$

上述表示关于 2，3 是反对称的，关于 4，5 也是反对称的。

它的中心 O 可以表示为

$$O(1,23,45):=23\wedge 45$$

它经过点 1 的直径可以表示为

$$\text{diameter}(1,23,45,1):=1(23\wedge 45)$$

它经过点 1 的切线可以表示为：

$$\text{hypertangent}(1,23,45,1):=[145]\sigma(3)12-[145]\sigma(2)13+$$

$$[123]\sigma(5)14 -$$
$$[123]\sigma(4)15$$

上述表示关于 2，3 是反对称的，关于 4，5 也是反对称的。

（4）在仿射平面上，经过 1，2，3 三点，并且具有一条渐近线为 45 的双曲线可以表示为

$$\begin{aligned}\text{hyperbola}(123,45,X):=\ &\sigma(12\wedge 45)^2[345][13X][23X]-\\&\sigma(13\wedge 45)^2[245][12X][23X]+\\&\sigma(23\wedge 45)^2[145][12X][13X]\\=\ &0\end{aligned}$$

上述表示关于 1，2，3 是反对称的，关于 4，5 也是反对称的。

它经过点 1 的切线可以表示为

$$\begin{aligned}\text{hypertangent}(123,45,1)=\ &\sigma(13\wedge 45)^2[245]12-\\&\sigma(12\wedge 45)^2[345]13\end{aligned}$$

上述表示关于 2，3 是反对称的，关于 4，5 也是反对称的。

它的中心 O 可以表示为

$$\begin{aligned}O(123,45)=\ &\sigma^2(1)[245][345](23\wedge 45)-\\&\sigma^2(2)[145][345](13\wedge 45)+\\&\sigma^2(3)[145][245](12\wedge 45)\end{aligned}$$

上述表示关于 1，2，3 是反对称的。

它经过点 1 的直径可以表示为

$$\begin{aligned}\text{diameter}(123,45,1)=\ &(\sigma^2(1)[345]^2-\\&\sigma^2(3)[145]^2)[245]12+\end{aligned}$$

$$(\sigma^2(2)[145]^2 -$$
$$\sigma^2(1)[245]^2)[345]13$$

上述表示关于2，3是反对称的，关于4，5也是反对称的。

它通过点1的直径的共轭直径可以表示为

$$\begin{aligned}\text{con-diameter}(123,45,1) = &\sigma^2(1)[145][245][345]23 + \\ &\sigma^2(1)[123][245][345]45 + \\ &\sigma^2(3)[145]^2[245]12 - \\ &\sigma^2(2)[145]^2[345]13\end{aligned}$$

上述表示关于2，3是反对称的，关于4，5也是反对称的。

(5) 在仿射平面上，经过1，2，3，4并且一条渐进线方向是T的双曲线可以表示为

$$\begin{aligned}\text{hyperbola}(1234,T,X) = &[12T][34T][13X][24X] - \\ &[13T][24T][12X][34X]\end{aligned}$$

上述表示关于1，2，3，4，T,X是反对称的。

它的中心O可以表示为

$$\begin{aligned}O(1234,T) = &a(b\sigma(2) - a\sigma(1))1 + \\ &b(a\sigma(1) - b\sigma(2))2 + \\ &c(a\sigma(1) + b\sigma(2))T\end{aligned}$$

这里如果用置换π作用，则置换前后只差一个置换符号。在此表示里面

$$a = [234][14T][23T]$$
$$b = [134][13T][24T]$$

$$c = [124][12T][34T]$$

对应上述（5）的表示，还可以给出经过 1、2、3 三点，以及两条渐近线方向 T_1 、T_2 的双曲线表示，它类似于上面的结果。值得注意的是，充分利用构造的条件以及射影的结果，将会使证明变得简洁快速，这在本章后面的例子中可以看到。

3.4 仿射二次曲线的构造与消元

本节给出的是仿射几何中特有的构造，省略了与射影几何相同的构造方式。

[CP－1（semi-free）]：X 是抛物线上的半自由点。

[CP－2（semi-free）]：X 是抛物线切线上的半自由点。

[CP－3－1（intersection）]：X 是一条直线与抛物线的交点。

[CP－3－2（intersection）]：X 是经过一个点且平行于抛物线的一条切线的直线与抛物线的交点。

[CP－3－3（intersection）]：X 是一条平行于已知直线的直线与抛物线的交点。

[CP－4－1（intersection）]：X 是一条直线与直径的交点。

[CP－4－2（intersection）]：X 是一条直线与过抛物线上一点的切线的交点。

[CP－4－3（intersection）]：X 是抛物线直径与抛物线上一

点的切线的交点。

[CP-4-4 (intersection)]: X 是一条抛物线切线与另外一条切线的交点，事实上就是两个切点连线关于抛物线的极点。

椭圆和双曲线的很多构造相同。另外由5点决定的椭圆双曲线和射影情况下相同。我们将省略相同的构造，换句话说，我们把目光集中在仿射几何特有的构造上。

下列构造是带有中心的二次曲线的构造，除了特别说明外，每一种构造都有三种情况：①由3点和中心 O 决定的二次曲线。此二次曲线可能是椭圆或者双曲线。②由3点和一条渐近线决定的双曲线。③由一点和两条渐近线决定的双曲线。如果渐近线给出的是方向，这和5点的情况完全相同。

[CC-1-1 (semi-free point)]: X 是二次曲线上一点。

[CC-1-2 (semi-free point)]: X 是双曲线的共轭双曲线上的一点。

[CC-2 (center)]: X 是二次曲线的中心。

[CC-3 (tangent point)]: X 是与一条切线平行的切线的切点。

[CC-4-1 (conic Intersection)]: X 是一条直线与二次曲线的交点。

[CC-4-2 (conic intersection)]: X 是与一条切线平行的直线与二次曲线的交点。

[CC-4-3 (conic intersection)]: X 是与一条直径平行的直线与二次曲线的交点。

[CC-4-4 (conic intersection)]: X 是一条通过双曲线上一点 A 的直线与其共轭双曲线的交点。

[CC－5－1 (line Intersection)]：X 是一条直线与一条切线的交点。

[CC－5－2 (line Intersection)]：X 是一条直线与一条直径的交点。

[CC－5－3 (line Intersection)]：X 是一条直线与一条直径的共轭直径的交点。

[CC－5－4 (line Intersection)]：X 是一条直线与双曲线一条渐近线的交点。

[CC－6－1 (diameter Intersection)]：X 是一条直径与一条渐近线的交点（O 点）。这是一个平凡构造。

[CC－6－2 (diameter Intersection)]：X 是一条直径与一条切线的交点。

[CC－6－3 (diameter intersection)]：X 是一条共轭直径与一条渐近线的交点（O 点）。这是一个平凡构造。

[CC－6－4 (diameter intersection)]：X 是一条共轭直径与一条切线的交点。

[CC－7－1 (self intersection)]：X 是两条切线的交点。

对应的构造也可以应用到双曲线上的共轭双曲线上。下面给出共轭双曲线的构造（图 3－1）：

$[723][745]\ \sigma^2(X)-[23X][45X]\ \sigma^2(7)$。相应的点按照如下结构进行构造：

6：$23\ \wedge 1\ \sigma$ (hypertangent(1,23,45,1))

7：$6\ \sigma\ (1O)\ \wedge O\ \sigma$ (hypertangent(1,23,45,1))

O：$23\ \wedge 45$

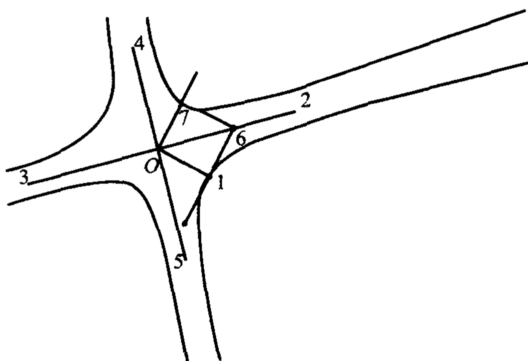


图 3－1　共轭双曲线的构造示意图

下面讨论二次曲线相关的消元问题。

消元规则 1：自由点　对于平面上多于 3 个的自由点，都可以用坐标化（扩张）的方法来进行消元。

消元规则 2：半自由点　对于二次曲线线上一点 X，设它满足 f_1，对于含有 X 的任意括号 $[EFX]$，仍然需要用坐标化的方法：选取 3 个自由点，然后对 f_1 进行扩张，结果仍然记为 f_1。同样对 $[EFX]$ 进行扩张，用 f_1 进行伪除。事实上，通常情况下，不需要进行半自由点的消去，需要的是对半自由点进行变换而达到证明的目的，称为仿射二次曲线变换。

消元规则 3：伪除消元算法　本方法对应于二次曲线部分的大部分构造，用特征列方法消元，虽然是完全的，但是比较复杂。本算法里面，维数设为 d，$f_1, f_2, \cdots, f_s$ 是 X 满足的括号多项式条件。现在要消去 P 中的 X，吴给出的算法如下。

步骤 1：选取 d 个自由点 $V_1, \cdots, V_d$，用此对 $f_i, i = 1, \cdots, n$，进行扩张操作（坐标化），然后收缩系数：

$$\{[A_{i_1}\cdots A_{i_{d-1}}X]: 1 \leqslant i_1 \leqslant \cdots \leqslant i_{d-1} \leqslant d\}$$

得到的结果我们仍然记为$f_1, f_2, \cdots, f_S$. 然后把括号作为主变量，计算特征列如下：

$$cs_1, \cdots, cs_t$$

步骤2：同样按照$A_1, \cdots, A_d$，对所有P中包含X的括号进行扩张，然后收缩系数，结果仍然记为P。

步骤3：将上述括号作为主变元，用求得的特征列$cs_1, \cdots, cs_t$对P进行伪除。

根据吴方法，非退化条件是伪除初式不为零，另外$[V_1\cdots V_d] \neq 0$。

利用此消元规则配合直线交点的消元规则可以涵盖全部的构造，但是这个过程是非常复杂的，有时候将括号写成行列式形式的多项式可能更好一些。事实上，仔细观察构造可以发现有如下几类。

一类是直线（一次）与二次曲线的交点，通常情况下我们用后面说到的仿射二次曲线变换以及带有约束的仿射二次曲线变换来完成。

另一类是直线与直线的交点，这些直线可能是切线、直径等，但是一般情况都是直线的线型结构，所以可以用直线交点的方法进行消元。

还有一类没有列出构造的是二次曲线与二次曲线的交点，通常情况下，都比较复杂，需要用上面的伪除方法，先求出特征列然后再消元。

在此，也可以提出仿射二次曲线变换：在按照上述构造的

二次曲线上任取一点，将满足此二次曲线的方程，这个方程一般都是单项式之和等于零的形式。我们将从一个单项式到其他单项式的转化称为仿射二次曲线变换，如果这种变换可以使多项式 P 降低项数或者次数。

例如：抛物线 parabola（123 T）上一点 X 满足：

$$[13X][23T][2XT]\sigma(1)-[13T][23X][1XT]\sigma(2)=0$$

则 $[13X][23T][2XT]\sigma(1)$ 到 $[13T][23X][1XT]\sigma(2)$ 的变换称为抛物线变换。如果一个点满足抛物线变换，则此点必然在某一条抛物线上。

在书后 5.5.4 节的双曲线例子中，最后消元就根据点在二次曲线上这个条件，也可以将它看做一种仿射二次曲线变换。也可以类比地定义有理二次曲线变换。

第4章
算法理论

本章对仿射括号代数计算中一些问题提出了相应的解决方法，提炼出来一些重要理论，同时给出了交换算法等一系列仿射括号代数的实用算法。

4.1 引　言

首先要比较一下仿射括号代数运算与射影里面括号代数运算，一般来说，前者要复杂一些。这种复杂的原因本质上是因为引入了新的变换，如果从仿射进展的模式，将会变得更复杂。

用仿射括号代数进行仿射几何计算是必要的，原因有如下几个方面。

(1) 仿射几何是射影几何的特例。所以很多射影几何中成立的命题、定理在仿射几何中自然成立。从理论上来说，在射影几何中证明了某一个命题也就等于在仿射几何中证明了它。

当然这需要对每一个命题都能知道它是属于射影几何的范畴还是仿射几何的范畴。但是作为一个完整的仿射几何计算系统，应该能够计算所有的仿射几何的例子，包括从射影几何来的例子。这就决定了计算范围的扩大，必须进行仿射括号代数的几何计算。

（2）另外很多仿射的命题不能退回到射影几何中，而这部分命题常常是仿射几何的精髓部分，这时用仿射括号就显得尤为必要了。

那么具体到计算层面上，为什么仿射括号代数要比括号代数困难呢？这主要是因为以下三个原因。

（1）σ-structure 的引入是为了保持计算的齐次性。但是一个向量（无穷远点）现在被转变成两部分：$\sigma(12) = \sigma(1)2 - \sigma(2)1$，增加了计算的复杂度。

（2）同样边界算子的引入带来了新的 syzygy 。对于计算来说，syzygy 库的增加，必然增加匹配和判断的困难。同样对括号多项式的收缩也变得困难，因为收缩时，除了括号存在可能，边界量部分也存在可能。

（3）仿射几何中更多、更复杂的表示从第 3 章就可以看出来。在射影几何里面二次曲线具有统一的形式，简单明了。但是在仿射几何里面，却有三种二次曲线，每一种还有不同的表示，有的表示还相当复杂。

由于仿射几何的本质，这三方面一般来说不能完全克服。但在实际中，我们可以有一些好的建议，这些建议将会大大降低上面三种情况所产生的困难。

我们作出如下建议。

（1）仅仅在恰当的时候才展开组合 σ 结构。这里所说的组

合 σ 结构 是指有两个向量在 σ 结构中。在二维仿射平面，仅有这一种形式，如果是 d 维仿射空间，就是指有 k 个向量在 σ 结构中，此时 $2 \leqslant k \leqslant d-1$ 。因为展开 σ 结构中会遇到项数迅速膨胀的问题。如果在一些适当的时机展开它将会避免这种情况，我们将在后面的章节里讨论这类问题。

（2）充分利用“无穷远”的概念。一条直线上的无穷远点可以表示为 σ（12）。在仿射二次曲线的表达中，用 T 或者 T_1 、T_2 等来表示。对于所有这些，如果再用边界算子作用，结果都将是零。所以在计算时要充分考虑到这一点。

（3）在实践中，对于一个括号多项式，减少 σ 结构中的向量数，会大大简化计算。比如对于一个多项式，尽管其各个单项式的 σ 结构中只含有 3 个不同的向量，这要比含有 4 个或者更多个不同的向量容易计算和收缩。我们提出的交换算法正是为了实现这个目的。

下面是计算中需要考虑的重要问题，它们的解决方法构成了本章的主要结构。

（1）在 4.2 节，我们试图经过交换的步骤处理括号和 σ 结构，这样使得消元变成一种统一的形式。我们称这种算法为交换算法。

（2）因为在处理 σ 结构时，不容易控制项数，所以在展开时必须根据具体情况进行控制和处理，这部分内容在 4.3 节有论述。

（3）4.4 节提到了算法实现过程中的其他一些问题，例如计算序、多项式形式变化、一些重要的公式以及多项式聚类问题。

（4）4.5 节我们给出了相关算法的描述。

4.2 交换算法

按照我们规定的序（参考4.4节），在 σ 结构中的向量多，整个多项式的序就高。这时我们就要改变向量所处的位置。另外在某些情况下，还需要把括号里面的某些向量移到 σ 结构里面。称这种向量位置的变换为交换，能够实现此目的算法称为交换算法。

首先，我们给出交换规则，在4.5节里面我们将给出交换算法。

交换规则1　通过逆用 syzygy 的法则有：

$$[123]\sigma(4) = [234]\sigma(1) + [124]\sigma(3) - [134]\sigma(2) \tag{4.1}$$

交换规则2

$$[123]\sigma(45) = [145]\sigma(23) - [245]\sigma(13) + [345]\sigma(12) \tag{4.2}$$

交换规则3　事实上，在d维空间有：

$$[c_1\cdots c_{d+1}]\sigma(a_1\cdots a_d) = \sum_{i=1}^{d}(-1)^{i+1}[c_i a_1\cdots a_d]\sigma(c_1\cdots\check{c}_i\cdots c_{d+1}) \tag{4.3}$$

上述3个公式把 σ 结构中的向量移动到括号中，下面我们对二维仿射平面的情形式（4.2）给出证明。

$$
\begin{aligned}
[123]\sigma(45) &= [123]\sigma(4)5 - [123]\sigma(5)4 \\
&= ([234]\sigma(1) - [134]\sigma(2) + [124]\sigma(3))5 - \\
&\quad ([235]\sigma(1) - [135]\sigma(2) + [125]\sigma(3))4 \\
&= ([234]5 - [235]4)\sigma(1) - ([134]5 - \\
&\quad [135]4)\sigma(2) + ([124]5 - [125]4)\sigma(3) \\
&= ([345]2 - [245]3)\sigma(1) - ([345]1 - \\
&\quad [145]3)\sigma(2) + ([245]1 - [145]2)\sigma(3) \\
&= ([345]2\sigma(1) - [345]1\sigma(2)) - \\
&\quad ([245]3\sigma(1) - [245]1\sigma(3)) + \\
&\quad ([145]3\sigma(2) - [145]2\sigma(3)) \\
&= [345]\sigma(12) - [245]\sigma(13) + [145]\sigma(23)
\end{aligned}
$$

交换规则 4 根据无穷远的表示 T 的特殊性质，有：

$$[12T]\sigma(3) = [13T]\sigma(2) - [23T]\sigma(1) \tag{4.4}$$

下面给出一个实际的例子：

例 如图 4－1 所示，抛物线点 1 处切线与点 3 处切线交于点 4，点 2 处切线与点 3 处切线交于点 5，点 1 处切线和点 2 处切线交于点 6，点 7 是过点 3 的直径与弦 12 的交点，则 47 平行于 62，57 平行于 61。

构造

自由点：1，2，3，T。

4：parabola（123T）在 1 处的切线与 parabola（123T）在 3 处的切线的交点。

5：parabola（123T）在 2 处的切线与 parabola（123T）在 3 处的切线的交点。

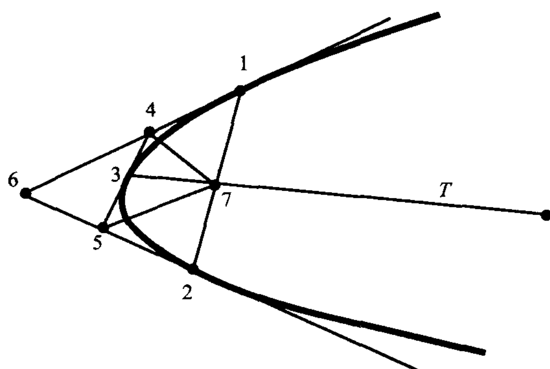


图4－1　交换算法例子的示意图

6：parabola（123T）在1处的切线与parabola（123T）在3处的切线的交点。

7：直线3 T与直线12的交点。

结论：47//62，57//61。

在消去7，6，5三点后，得到如下结果：

$$[13T]^2\sigma^2(2) - [12T]^2\sigma^2(3) + [23T]^2\sigma^2(1) - 2[13T][23T]\sigma(1)\sigma(2)$$

如果用上面总结出来的类余弦公式来收缩，马上可以收缩到零。但这是我们总结出来的规则，要求库里面含有对应的规律。而这样的问题如果能在计算部分解决将会大大降低最后匹配的困难。如果没有交换算法，计算机在计算规则部分将用grammer法则对某个单项式第一个括号展开，大致需要使用三次这样的计算规则才能达到我们的目的。如果我们把交换算法加入进来，在每一次计算中，程序将判断是不是

需要进行交换。根据上面的序，程序将对 $[12T]\sigma(3)$ 进行作用，目的就是要把 $\sigma(3)$ 全部移到括号中去，此时马上就可以得出结果零。

在众多的计算例子中，我们发现类似的情况很多。例如在计算抛物线例子某一步中遇到：

$$-[125][13T]\sigma(6)+[235][16T]\sigma(1)-[36T][125]\sigma(1)$$

如果不进行交换，消元 6 点时，必须计算 $\sigma(6)$，$[16T]$,$[36T]$ 三个括号。但是如果把 6 移到括号中，我们仅需要计算 $[16T]$,$[36T]$。类似的例子很多。当然在某些情况下，经过交换多项式项数会增加，但是这根本不会增加难度，因为通常情况下，这种情况会带来消去 GCD 或者能够进行分解的好处。

事实上，进行交换是必要的，原因有如下几个方面。

(1) 它有益于程序进行系统实现。在引入交换算法之前，实现仿射括号代数计算，必须考虑消元的点在什么结构里面，不同的结构需要用不同的展开。这两种不同的处理方式带来了计算方法上的困难。我们也不能直接把括号代数一些算法拿过来，但是在引入交换算法后，这些都可以统一为对括号结构的消元。

(2) 把向量从 σ 结构移到括号中，因为括号中还有其他的向量，这将可能带来更简化的展开形式。

(3) 对于任何多项式，如果其 σ 中的向量的种数越少，则提取公因子和各单项相互合并抵消的可能性就越大。

4.3 括号结构与σ结构

在本节中，#操作符表示一个集合的元素个数。同时本节中的“最小展开”，是指具有最低计算序的展开（参考4.4节）。

命题4.3.1 首先设{1，2，3，4}是仿射平面的任意4点，考虑表达式$[12\sigma(34)]$，设$P=\{1,2\}$，$Q=\{3,4\}$，上述表达式具有如下性质：

（1）（平凡为零）如果$\#(P\cap Q)=2$，平凡为零。在下列情况，默认这种情况不出现。

（2）（单项展开）它有单项展开当且仅当$\#(P\cap Q)=1$。例如：

$$[12\sigma(14)]=-[124]\sigma(1)$$

（3）（其他情况）它有展开：$[12\sigma(34)]=[124]\sigma(3)-[123]\sigma(4)$。它的最小展开为：$[134]\sigma(2)-[234]\sigma(1)$。

命题4.3.2 设{1，2，3，4，5}仿射平面的5个点。我们考虑表达式$[1\sigma(23)\sigma(45)]$的展开情况，设$P=\{2,3\}$，$Q=\{4,5\}$，上述表达式具有如下性质：

（1）（平凡为零）如果下列一个条件满足，则表达式平凡为零：①P＝Q；②2＝3；③4＝5。

在下述情况，我们将不考虑这种平凡为零的情况。

（2）（单项展开）如果$\#(P\cap Q)=1$，它有单项展开。例如：

$$[1\,\sigma(23)\,\sigma(25)] = -[235]\,\sigma(2)\,\sigma(1)$$

(3)（两项展开）否则它有两项展开。这个通过下列公式得到：

$$[1\,\sigma(23)\,\sigma(45)] = \sigma(1)\,\sigma(23 \wedge 45)$$

它的最小展开为

$$[245]\,\sigma(3)\,\sigma(1) - [345]\,\sigma(2)\,\sigma(1)$$

命题 4.3.3 设 $\{1, 2, 3, 4, 5, 6\}$ 是仿射平面的 6 个点。我们考虑表达式 $1\,\sigma(23) \wedge 4\sigma(56)$ 的展开情况：

(1)（平凡为零）如果下列的一个条件满足，上述表达式平凡为零：①$1=4$ 并且 $\{2, 3\} = \{4, 5\}$；②$2=3$；③$5=6$。

在下列情况，假定表达式不平凡为零。

(2)（单项展开）如果下列情况之一满足，表达式由单向展开：

情况 A：$1=4$ 并且 $\#(\{2,3\} \cap \{5,6\}) = 1$。例如为

$$1\,\sigma(23) \wedge 1\,\sigma(26) = -[236]\,\sigma(1)\sigma(2)\,1$$

情况 B：$1 \in \{2,3\}$ 并且 $4 \in \{5, 6\}$ 并且 $1 = 4$。例如为

$$1\,\sigma(13) \wedge 1\,\sigma(16) = -[136]\,\sigma^2(1)\,1$$

情况 C：$1=2$，$3=4 \in \{5, 6\}$ 或者 $1 = 3, 2=4 \in \{5, 6\}$。例如为

$$1\,\sigma(13) \wedge 3\,\sigma(36) = -[136]\,\sigma(1)\,\sigma(3)\,3$$

情况 D：$4=5, 1=6 \in \{2, 3\}$ 或者 $4 = 6, 1=5 \in \{2, 3\}$。

(3)(两项展开) 如果下列条件之一满足，表达具有两项展开：

情况 A：1 = 4。例如：$1\sigma(23) \wedge 1\sigma(56) = -\sigma(1)\sigma(23\wedge 56)1$。

它有最小展开：

$$[356]\sigma(1)\sigma(2)1 - [256]\sigma(1)\sigma(3)1。$$

情况 B：1 = 2，4 = 3 或者 1 = 3，4 = 2。

情况 C：4 = 5，1 = 6 或者 4 = 6，1 = 5。例如为

$$1\sigma(14)\wedge 4\sigma(56) = -\sigma(14\wedge 56)\sigma(1)4$$

它有最小展开：

$$([456]\sigma(1) - [156]\sigma(4))\sigma(1)4$$

情况 D：$1\in\{2,3\} = \{5,6\}$。例如为

$$1\sigma(13)\wedge 4\sigma(13) = [134]\sigma(1)\sigma(13)$$

它有最小展开：

$$[134](\sigma^2(1)3 - \sigma(1)\sigma(3)1)$$

(4)(三项展开) 如果下列条件满足：① $1 = (\{2,3\}\cap\{4,5\})$；② $4 = (\{2,3\}\cap\{4,5\})$。例如：

$$1\sigma(13)\wedge 4\sigma(16) = -[134]\sigma(16)1 + [136]\sigma^2(1)4$$

(5)(四项展开) 如果下列两组条件之一满足，表达式具有四项展开：

情况 A：$1 \in \{2,3\}$ 或者 $4 \in \{5,6\}$

情况 B：$1 \in \{5,6\}$ 或者 $4 \in \{2,3\}$

(6)(五项展开) 如果 $\#(\{2,3\} \cap \{5,6\}) = 1$，它有五项展开。

(7)(其他情况) 它最少有 6 项展开。

上述命题的证明是显然的。用 3.1，3.2，3.3 节内容我们可以相应地展开下述这些情况：

$$12 \wedge 3\,\sigma\,(45) \wedge 6\,\sigma\,(78)$$

$$1\,\sigma\,(23) \wedge 4\,\sigma\,(56) \wedge 7\,\sigma\,(89) \qquad (4.5)$$

4.4 仿射括号代数算法的其他问题

4.4.1 规则形式与计算序

首先，我们希望系统对每一步的计算结果都给出一个具有特定形式的多项式。这个特定形式通过如下的定义实现。

定义 4.4.1 规则多项式(the polynomial with normal form)：在仿射括号代数中，一个多项式被称为是规则形式的，它要满足如下两个条件。

(1) 仅仅包含括号结构、σ 结构、∧ 结构三种结构，并且外积结构是指能够转化成括号的外积结构。在 2-D 仿射平面上，仅有一种形式 $12 \wedge 34 \wedge 56$。

（2）所有项已经按照定义4.4.2的序排好顺序。通常情况下，有一些例外可能会超出以上的多项式范围，例如证明一个点等于另外一个点，这时我们考虑它们的系数。

定义4.4.2 我们给出的计算序的定义如下：

（1）定义两个向量具有 $V_i < V_j$，如果 V_j 在 V_i 后被构造。如果 V_i 和 V_j 同时被构造，换句话说，它们之间没有依赖关系，此时比较它们的次数，次数低的序高。如果仍然不能区分，只需要按照字母表排序。

下面假设已经排好序：$V_1 < \cdots < V_n$。

（2）括号里面向量的序是指括号里面的向量已经按照上述序排好。下面假定所有括号里面的向量都已经排序好。

（3）两个括号的序：具有次数 i 的括号 A 比 次数为 j 的括号 B 序低即 $A < B$，如果 $i < j$。如果 $i = j$，仅需要比较它们在上述序诱导下来的字典序。

（4）关于 σ 结构 的序定义类似，同类结构序的定义类似。

（5）不同结构的序：括号结构 $<$ σ 结构 $<$ 外积结构。

（6）单项式序的定义为在上述序诱导下的字典序。

定义4.4.3 一个括号多项式被称为是完全展开的（the polynomial with complete expansion）是指它满足如下两个条件：

（1）它仅仅包括括号和 σ 结构并且后者结构里只含有一个向量，例如，$[123]\sigma(4)$，$[12]\sigma(3)\sigma(4)$。

（2）在上述序下已经排好序。

因为syzygy的存在，多项式形式有很多种可能。虽然它们是相等的，但是不同的形式增加了处理的难度。一种形式在一个序下好，并不代表在所有序下都好。

定义 4.4.4 order-list：对于一个多项式，定义它的 order-list 为如下形式：

[numa(P),1/numg(P),nums(P),numi(P)]

这里 numa(P) 表示它含有构造的向量的个数。重复的不计入。如果所有的向量都是自由的，那么 numa(P) =0。

numg(P) 表示 P 的 gcd 的括号次数。一个括号就记为 1，一个 σ 结构也记为 1。例如，P 具有 gcd： = 4 [12] [123] σ(2) σ (3)，则有 numg(P) =4。

nums(P) 表示在 σ 结构中向量的个数，重复的不计入。

numi(P) 表示 P 中单项式的数目。

这样可以按照上面的 order-list 给出多项式的序。我们希望在计算时给出 order-list 最低的多项式（the polynomial with lowest order）。图 4 - 2 给出了它们之间的相互关系。

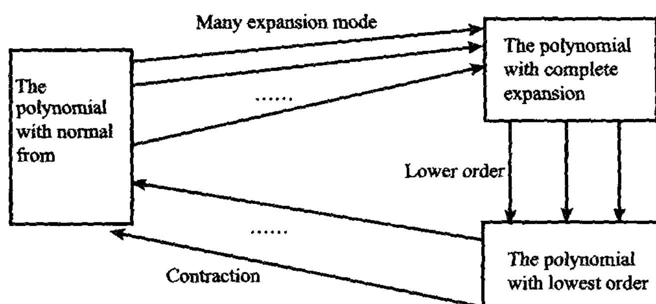


图 4 - 2 不同多项式之间的转换

为了降低 numi(P)，我们将用后面的收缩算法，但是收

缩可能会增加 nums(P)；为了降低 nums(P)，我们使用交换算法，但是交换可能会增加 numi(P)。在推理过程中，我们可能需要因式分解更优先，以便获得最终的推理结果，这时我们只需要简单地调整 order-list 就可以了。

一个括号多项式的最小展开，是指在上述序下此多项式序最低的展开。

4.4.2 一些常用的公式及其扩展

仿射括号代数的计算离不开有效的公式。一般从仿射括号代数的 syzygy 中，可以得出很多公式。

但是它们中的一些可能很少用到。这里我们仅给出一些常用的重要公式。通过使用这些公式，将会大大简化仿射括号代数的计算。

(1) 在二维仿射平面：

$$[12\sigma(34)] = \sigma(12 \wedge 34) \tag{4.6}$$

在 d 维仿射空间中，如果 $k+j=d+1$，我们就有

$$[a_1\cdots a_k\,\sigma(b_1\cdots b_j)] = \sigma(a_1\cdots a_k \wedge b_1\cdots b_j) \tag{4.7}$$

(2) 在二维仿射平面上：

$$[1\,\sigma(23)\,\sigma(45)] = \sigma(1)\,\sigma(23 \wedge 45) \tag{4.8}$$

在三维空间中：

$$[12\,\sigma(34)\,\sigma(56)] = -\sigma(1)\,\sigma(234 \wedge 56) + \sigma(2)\,\sigma(134 \wedge 56) \tag{4.9}$$

在 d 维空间中，我们有：

$$[a_1\cdots a_{d-1}\sigma(12)\sigma(34)] =$$

$$\sum_{i=1}^{d-1}(-1)^{i-1}\sigma(a_1\cdots\check{a}_i\cdots a_{d-1}\ 12\wedge 34) \qquad (4.10)$$

证明：

这些公式的证明类似，我们给出其中式（4.9）的证明：

$$[12\ (\sigma(3)4-\sigma(4)3)(\sigma(5)6-\sigma(6)5)]$$

$$=[1246]\sigma(3)\sigma(5)-[1236]\sigma(4)\sigma(5)-$$

$$[1245]\sigma(3)\sigma(6)+[1235]\sigma(4)\sigma(6)$$

$$=([1246]\sigma(3)-[1236]\sigma(4))\sigma(5)-$$

$$([1245]\sigma(3)-[1235]\sigma(4))\sigma(6)$$

$$=([2345]\sigma(1)\sigma(6)-[2346]\sigma(1)\sigma(5)+$$

$$[1346]\sigma(2)\sigma(5)-[1345]\sigma(2)\sigma(6)$$

$$=-\sigma(1)\sigma(234\wedge 56)+\sigma(2)\sigma(134\wedge 56)$$

（3）在三维空间中：

$$[1\ \sigma(23)\ \sigma(456)]=\sigma(1)\ \sigma(23\wedge 456) \qquad (4.11)$$

这些等式可以被扩展到更高维空间。在 d 维空间中我们有如下的公式：

$$[a_1\cdots a_{k_1}\sigma(b_1\cdots b_{k_2})\ \sigma(c_1\cdots c_{k_3})]$$

$$=\sum_{i}^{k_1}(-1)^{i+1}\sigma(a_i)$$

$$\sigma(a_1\cdots\check{a}_i\cdots a_{k_1}b_1\cdots b_{k_2}\wedge c_1\cdots c_{k_3}) \qquad (4.12)$$

这里，$k_1\leqslant d$，$k_2\leqslant d$，$k_3\leqslant d$，以及 $k_1+k_2+k_3=d+3$

（4）类余弦收缩公式：在二维仿射平面中，T 作为无穷远点引入，具有性质 $\sigma(T)=0$，有如下漂亮形式的收缩公式，它将三项收缩为一项：

$$[12T]^2\sigma^2(3)+[13T]^2\sigma^2(2)-[23T]^2\sigma^2(1)$$
$$=2[12T][13T]\sigma(2)\sigma(3) \qquad (4.13)$$

$$[12T]^2\sigma^2(3)+[23T]^2\sigma^2(1)-[13T]^2\sigma^2(2)$$
$$=-2[12T][23T]\sigma(1)\sigma(3) \qquad (4.14)$$

$$[13T]^2\sigma^2(2)+[23T]^2\sigma^2(1)-[12T]^2\sigma^2(3)$$
$$=2[13T][23T]\sigma(1)\sigma(2) \qquad (4.15)$$

证明：三个公式的证明类似，是很容易证明的。

$$[12T]^2\sigma^2(3)+[13T]^2\sigma^2(2)-[12T][13T]\sigma(2)\sigma(3)$$
$$=([12T]\sigma(3)-[13T]\sigma(2))^2$$
$$=[23T]^2\sigma^2(1)$$

4.4.3 多项式聚类

在计算过程中，很多时候需要的并不是同时对 P 的所有单项式进行操作，而是对一部分单项式进行操作。换句话说，我们希望对括号多项式的所有单项式进行分类，相似性高的放在一起，便于进行分解和操作，我们称此为单项式聚类。

单项式聚类通过寻找各个单项式之间的相似性来实现。这需要定义不同的单项式之间的相似性。在数据挖掘的聚类算法中，大致有距离方法、划分方法、层次方法、基于密度方法、基于网格方法和基于模型方法这6种常用的方法[5]，当然可以

把这些方法都用在单项式聚类上。

我们使用的是距离方法。首先假定所有的单项式中的括号以及括号中的向量已经排好序。

定义一个单项式的指标分解是指对于单项式 M 中存在重复的向量 V，如果出现 k 次，则将各个括号中的 V，按顺序分别用 $V_1, V_2, \cdots, V_k$ 代替，这称为单项式的指标分解，我们记分解后的单项式为 M^* 。这样通过指标分解，我们仅需考虑多重线性的单项式即可。以下都是针对多重线性的单项式。

首先定义向量 V 在单项式 $M = B_1 \cdots B_n$ 中的坐标。设 $B_i = [X_{i1} \cdots X_{id}], 1 \leqslant i \leqslant n$ ，对 M 写成矩阵形式如下：

$$\begin{pmatrix} X_{11} & \cdots & X_{1d} \\ \vdots & & \vdots \\ X_{n1} & \cdots & X_{nd} \end{pmatrix}$$

我们把向量 V 在此矩阵中的行列坐标称为 V 在单项式 M 中的坐标，记为 $\mathrm{cor}(V,M)$。例如对于上述的表示方法 $\mathrm{cor}(X_{ij},M) = (i,j)$ 。

对于向量 V，设 $\mathrm{cor}(V,M_1) = (x_1,y_1), \mathrm{cor}(V,M_2) = (x_2,y_2)$，则 V 在两个单项式中的距离为

$$\mathrm{dis}(V,M_1,M_2) = |x_1 - x_2| + |y_1 - y_2|$$

同样定义两个单项式的距离为

$$\mathrm{distance}(M_1,M_2) = \sum_{V \in \mathrm{elements}(M_1)} \mathrm{Dis}(V,M_1,M_2)$$

通过此，我们就可以通过计算单项式之间的距离进行聚

类，然后针对同一类的单项式进行 GCD 的提取和分解。对于仿射括号代数的计算，我们把 σ 结构看做括号，用上面的方法进行聚类。

4.5 相关算法

4.5.1 算法1：交换算法

输入：括号多项式 P

输出：交换后的多项式

对于 P 中的每一个单项式，存在下面所列举的可能。在每一个单项式被处理后，我们将计算并输出结果。下列序是按照消元（或者说构造）序排列的，并且假定所有相关的已经排好序。

situation A：如果单项式具有形式：$[\boldsymbol{V}_1\boldsymbol{V}_2\boldsymbol{V}_3]\ \sigma(\boldsymbol{V}_4)$。如果 $\boldsymbol{V}_4$ 的序比任何 $\boldsymbol{V}_1,\boldsymbol{V}_2,\boldsymbol{V}_3$ 中的向量的序高，我们用交换规则把 V_4 移到括号内部。

situation B：如果单项式具有形式 $[\boldsymbol{V}_1\boldsymbol{V}_2\boldsymbol{V}_3]\ \sigma(\boldsymbol{V}_4)\cdots\sigma(\boldsymbol{V}_k)$，首先我们移动具有更高序的向量，然后按顺序循环。

situation C：如果单项式具有形式 $[\boldsymbol{V}_1\boldsymbol{V}_2\boldsymbol{V}_3][\boldsymbol{C}_1\boldsymbol{C}_2\boldsymbol{C}_3]\ \sigma(\boldsymbol{V})$，我们比较字典序 $[\boldsymbol{V}_1\boldsymbol{V}_2\boldsymbol{V}_3]$ 和 $[\boldsymbol{C}_1\boldsymbol{C}_2\boldsymbol{C}_3]$，然后选择具有低序的括号与边界算子组合，例如 $[\boldsymbol{V}_1\boldsymbol{V}_2\boldsymbol{V}_3]$。应用 situation A 于 $[\boldsymbol{V}_1\boldsymbol{V}_2\boldsymbol{V}_3]\ \sigma(\boldsymbol{V})$。如果有多个括号，我们选择序最

低的。

situation D：如果单项式具有多个括号和多个 σ 结构，则可以分解为 situation B 和 situation C 来处理。

4.5.2 算法2：收缩算法

这里我们首先建立一个序，在这个序下相邻的两个单项式最有可能收缩，然后我们再用组合的办法处理。设 num(V , P) 用来表示 P 中 V 的个数，num(σ,P) 用来表示 P 中 σ 的个数，num(B , P) 用来表示 P 中括号 B 的个数，num($\sigma(V)$,P) 用来表示 $\sigma(V)$ 在 P 中的个数。

定义收缩用的序如下：

(1) 对于任意 P 中的 V_1,V_2，如果 $\text{num}(V_1,P) \leqslant \text{num}(V_2,P)$，定义 $V_1 < V_2$。下面假定 P 中所有的向量已经排好序并且记录为：$V_1 < \cdots < V_n$。

(2) 对于两个括号 $B_1 = [V_{i_1}\cdots V_{i_d}], B_2 = [V_{j_1}\cdots V_{j_d}]$，如果 $B_1 < B_2$ 是指它们满足如下条件之一：

situation A：$\text{num}(B_1,P) < \text{num}(B_2,P)$。

situation B：如果 $\text{num}(B_1,P) = \text{num}(B_2,P)$，

定义 $[i_1,\cdots,i_d] <_{\text{lex}} [j_1,\cdots,j_d]$。

(3) 对于两个 $\sigma(V_1),\sigma(V_2)$，定义 $\sigma(V_1) < \sigma(V_2)$，如果：

situation A：$\text{num}(\sigma(V_1),P) < \text{num}(\sigma(V_2),P)$。

situation B：如果 $\text{num}(\sigma(V_1),P) = \text{num}(\sigma(V_2),P)$，定义 $V_1 < V_2$。

(4) 对于括号 $B = [V_1\cdots V_d]$ 和 $\sigma(V)$，定义 $B < \sigma(V)$。

如果：

situation A: $num(B,P) < num(\sigma(V),P)$

situation B: $max(num(V_1,P),\cdots,num(V_d,P)) \leqslant max(num(B,P),num(\sigma,P))$。

(5) 这样在上述基础序的定义下我们可以定义 P 的所有单项式序，它就是上述序诱导下的字典序。

收缩算法：

输入：待收缩的多项式 P，它含有 n 个单项式。

输出：收缩后的多项式 P，它含有少于或者等于 n 个多项式。

begin: temp: = p, tempnew: = p。

步骤 1：利用上述序的定义，对 P 进行排序；

步骤 2：执行收缩算法在相邻的两个单项式，并且记录最后的结果为 tempnew。

步骤 3：如果 tempnew = temp 转步骤 4；否则 temp: = tempnew，转步骤 1。

步骤 4：执行整个多项式中任意两项间的收缩。结果仍然记为 tempnew。

步骤 5：如果 tempnew = temp 转步骤 6；否则 temp: = tempnew 转步骤 1；

步骤 6：输出 tempnew。

在一些情况下，收缩通常需要先将一项展开成更多的项，然后才能进行有效的收缩，这个过程并不好控制。通常对于那些正常算法不能收缩的情况，采取对其中某一项进行展开（这个过程称为爆炸），然后对新的多项式进行收缩的方法来

处理。同时对多项爆炸然后收缩的情况也是存在的，但是很难处理。

4.5.3 算法3：GCD提取算法

实现最简单的GCD提取算法，我们可以借助于多项式的成熟方法，通过对括号多项式的多项式化，就可以利用多项式的GCD提取算法。这种变换不需要坐标化。这里说的GCD当然是对括号多项式的各个单项说的，也就是说GCD最多是个单项式。

当然如果这样做并不是完全的，因为虽然表面上可能没有GCD，但是因为syzygies的存在，经过变换可能就有了。

利用第2章的整除性理论，我们可以很好地解决这个问题。

但关键问题是，我们需要找到那个试探用的单项式。我们通过查找一个一个的括号来进行此过程。

判断什么样的括号能整除一个多项式，我们可以从三个方面考虑这种可能性。

（1）一个是此括号在P中出现的次数，一般情况出现的次数越多，可能性越大。

（2）与此括号相联系的syzygy的其他括号在多项式中出现的次数，一般也是出现次数越多，可能性也就越大。

（3）一个括号完全没有出现，这种可能性或者极大或者极小。

输入：包括n项的括号多项式P，syzygies库Γ。

输出：P的GCD。

begin temp: $=P, k=1$, gcdp: $=1$

步骤1：对可能的括号按照可能性排序，$i=1$。

步骤2：用整除性判定算法，判定第 i 个括号 B_i 是否能整除 P，如果 i 达到最大值，转步骤5。

步骤3：如果能，转步骤4，否则 $i:=i+1$，转步骤2。

步骤4：P 去除一括号 B_i，仍然记为 P，gcdp: $=$ gcdp $*$ B_i。如果 P 为常数转步骤5，否则转步骤1。

步骤5：输出 gcdp，程序终止。

4.5.4 算法4：因式分解

多项式的因式分解要比提取 GCD 更复杂，因为可能性也更多，通常我们给出的算法如下：

输入：括号多项式 P，*syzygy* 库 Γ。

输出：P 的分解结果。

步骤1：对 P 基于括号进行多项式化，并利用多项式分解算法进行分解。

步骤2：对 P 的各项进行组合，然后利用 GCD 提取方法，进行部分 GCD 的提取。

步骤3：对每个组合部分的商进行收缩，并全部按照特定序展开，看是否相等。

步骤4：如果相等，则进行分解，否则继续进行试探。

4.5.5 主算法1：自动证明

输入：多项式 P，构造的点 $[V_1,\cdots,V_n]$，点的构造表达

[exp(V_1),…,exp(V_n)]。

输出：消去点后的表达式，如果为零则命题得证，同时产生证明过程的 GCD。

开始：temp：=p,i=1，gcdp：=1。

步骤 1：重新判定消元顺序，如果判定结果与输入的顺序不同，给出提示，等待操作（可以跳过）。

步骤 2：temp：=eliminate（i，temp），如果 temp=0 转步骤 6，否则转步骤 3。

步骤 3：进行交换算法，则 temp：=exchange（temp），如果 temp=0 转 步骤 7，否则进行步骤 4。

步骤 4：进行 GCD 提取和剔除操作，temp：=removegcd（temp），gcdp：=gcdp × getgcd（temp），然后进行步骤 5。

步骤 5：进行收缩算法，temp：=contract（temp），如果 temp=0 转步骤 7，否则转步骤 6。

步骤 6：类似于步骤 4，再次进行 GCD 提取和剔除操作，置 temp：=eliminategcd（temp）以及 gcdp：=gcdp × getgcd（temp），i = i +1，转步骤 2。

步骤 7：程序终止并输出 result 和 gcdp。

4.5.6 主算法 2：带有目的性的自动推理

输入：多项式 P，目标多项式描述 Q，构造的点 [V_1,…,V_n] 以及点的构造表达 [exp(V_1),…,exp(V_n)]。

输出：P 的带有 Q 的分解形式。

开始：temp：=p,i = 1。

步骤 1：与 Q 比较，确定消去点的集合。

步骤2：重新判定消元顺序，如果判定结果与输入的顺序不同，给出提示，等待操作（可以跳过）。

步骤3：类似 myprover 的消元过程，如果结果为零转步骤7，否则继续。

步骤4：关于 Q 对 P 进行多项式整除性判定。如果能整除转下一步，否则转步骤7。

步骤5：设分解后的商为 S，对 S 进行收缩和 GCD 处理，结果仍然记为 S。

步骤6：输出 S 和 Q。

步骤7：输出结果为零，或者不能分解。

4.5.7 主算法3：自动推理

输入：多项式 P，构造的点 $[V_1,\cdots,V_n]$，点的构造表达 $[\exp(V_1),\cdots,\exp(V_n)]$。

输出：P 的分解形式。

开始：temp：$=p, i=1$。

步骤1：重新判定消元顺序，如果判定结果与输入的顺序不同，给出提示，等待操作（可以跳过）。

步骤2：进行类似于 myprover 的点的消除，记消去点的结果为 temp，如果点都消去结束，转步骤6。

步骤3：对 temp 进行分解，分解结果的项记入到数组 result 中，转步骤4，否则转步骤2。

步骤4：对 result 数组的每个元素进行步骤5。

步骤5：对应 result 数组的每一项进行尝试收缩和 GCD 提取。

步骤6：如果 result 中只有一项（没有分解成功）给出没有成功的提示，否则给出分解结果。

当然具体实现时，3 个主算法用一个函数就可以了，执行哪一个算法过程可以通过输入参数的不同形式来控制，同时主程序的输出信息应该是可以调整的，根据不同的需要调整为不同的输出。

第5章
自动证明应用与实现

本章介绍了仿射括号代数计算在自动证明中的应用，并对系统进行了介绍和讨论，摘录了16个典型例子，同时还给出一些关键性的代码。

5.1 消元顺序

一般来说，应用仿射括号代数进行几何计算或者几何定理机器证明都要经过如下几个机械化的步骤。

（1）表示：表示就是将几何代数化，这是整个计算系统的入口。对于一个现实的系统来说，直接从几何描述或者示意图来进行计算是非常不现实的。几何代数化的过程就是把各种几何构造转化为代数表示，这里面就有一个表示的多样性问题。

(2) 消元：消元就是把构造的点逐步消去。在完成了上述表示之后，其实就是把一个表达式里面的点用另外一个表达式代替。但是往往用来代替的表达式具有不同的形式，这时选择的标准通常是使代换后的表达式简单，比如有公因子、项数较少等。

(3) 展开：将几何信息转化为代数表示，通常都是用 Grassmann-Cayley 代数。众多 Grassmann-Cayley 代数表示到括号代数还要经过展开的步骤，从前面第 2 章的介绍可以知道，通常展开都不是唯一的。这个过程一般伴随在消元过程中，并对定理证明、几何计算的难易产生很大影响。

(4) 化简：一般仿射括号代数中的化简也包括收缩（contraction）、合并（conbination）、分解（factorization）。收缩是指尽量减少括号多项式的项数；合并是上述展开的逆；分解是指模去仿射括号代数不变量环的 syzygy 理想，也就是用 syzygy 进行化简。

根据以前的章节，我们可以在程序里面实现这些过程，同时我们的程序还进行一项系统的判定任务，那就是消元顺序的重新排列。从理论上说，不同的消元顺序其结果应该都是一样的，但是对于程序执行的效率是大大不同的，中间展开的项数也是变化很大。程序按照输入的构造顺序进行消元（逆过程）并不一定是最有利的。

我们把构造一个点所需要的其他的点，称为此点的构造基集合，用记号 $C(A)=\{\boldsymbol{B}_1,\cdots,\boldsymbol{B}_k,\}$ 表示 A 的构造基于 $\boldsymbol{B}_1$，…，$\boldsymbol{B}_k$。

我们把构造顺序分成以下三类。

第一类是堆砌式。堆砌式的构造方式是所有的构造点都是由固定的一些自由点和半自由点构造的。

第二类是连锁式。连锁式的构造方式是后面的构造点利用自由点、半自由点还有前面的构造点一起通过几何结构而构造的。

第三类是上述两者的混合。某些构造点是基于堆砌式，其他一些构造点是基于连锁式。

一般来说，一个具体命题的构造方式都可以用一个图来表示，我们称为构造图。图5-1给出了例5.6的构造图。

通常构造图很杂乱，用在程序中也不现实。在实际中我们可以用其他方法来代替。下面给出了使用List的方式来表示第5.5节例5.7的构造图，我们实现的程序就是用这种方法：

[1,2,3,4] ⟶ 自由点

[[1,2,4],[5]] ⟶ 半自由点

[[1,3,4],[6]] ⟶ 半自由点

[[1,3,4,6],7] ⟶ 构造点

[[2,3,5,6],8] ⟶ 构造点

[[1,2,4,5],9] ⟶ 构造点

[7,8,9] ⟶ 结论

我们首先看一下连锁式的构造顺序，这种构造顺序一般来说是没有必要调整的，因为通过构造关系基本上已经给定了一个固定的序，只有在此序下消元才能保证结果的完全

性。但是也有一类情况，那就是对于某些构造点，它们有共同的构造基集合。这时我们将用后面提到的方法进行调整。

对于堆砌式的构造方法，一般来说按照构造点、半自由点、自由点的大框架顺序进行消元都没有问题，但是不同的顺序会导致不同的复杂程度。

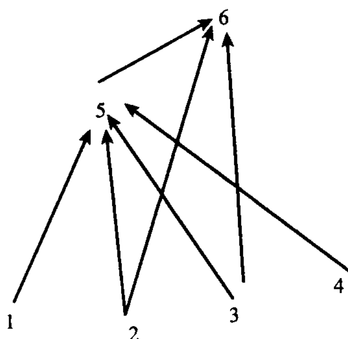


图 5-1　例 5.6 的构造图

对于第三类，一般通过先消去连锁方式生成的构造点，然后再消去基于自由点或半自由点构造的构造点，最后是半自由点。

具体来说，对于堆砌式的构造方法，通过如下的方式来确定消元顺序。对于两个构造点 V_1 和 V_2。

（1）如果 $C(V_1) \subseteq C(V_2)$，则 V_2 在 V_1 之前消去。

（2）如果 $C(V_1) = C(V_2)$，如果 V_1，V_1 中某一个只分布在一个结构（这里的结构是指括号结构或者 σ 结构），则此向量后消去。

(3) 否则比较 V_1 和 V_2 所对应的 $C(V_1)$,$C(V_2)$ 的点多少，点多的先消去，这是为了使展开时项数最少。

(4) 否则比较 V_1 和 V_2 的次数，次数低的先消，并且注意在消元时尽量先进行交换，达到统一结构，然后再消去。

对于半自由点的消去，应该是高次约束的后消，低次约束的先消去。

最后我们还要根据结论的表述，进一步讨论消元的顺序问题。

结论的表述有两种可能，一种是完全由连锁式构造的点构成，另外一种就是由自由点、半自由点加上构造的点组成，显然肯定含有最后构造的点。

对于第二种，通常按照我们构造顺序加上上面考虑的顺序进行消元，都可以满足需要。在进行消元计算时，展开过程中要充分利用每一步的自由点、半自由点，使得展开形式简单。

对于第一种，也完全可以按照上述方法进行消元，只是通常情况下，很难利用自由点、半自由点，但可以利用前面构造点来使展开简化。

对于结论中还有一类，就是里面构造大多数都是堆砌式的，但是结论表述都是或者大多数是关于构造点的，这种结论的计算，因为结论中相互点之间没有明确的关系，没有办法利用来简化展开。但是可以使用这样的方法：选取构造基集合中点的数目最少的点，然后先对其进行消去，消去后的结论将包含部分自由点、半自由点，然后再进行正常的消元算法。

5.2 系统说明与具体实现

我们实现系统的环境是 Maple 10，虽然本质上在任何环境中实现都不是困难的。事实上在 Maple 10 环境中，因为括号代数本质上与其他计算不同，我们几乎没有利用任何 Maple 本身的函数以及计算优势，从底层建立了整个仿射括号代数的计算系统。

我们在程序中实现了所有的函数和相关算法。另外需要说明的是：虽然整个系统是为了进行二维仿射括号代数计算，但是从一开始我们就考虑了它的扩展性，在建立数据结构中，我们给出了维数的控制部分，它默认的是二维，但是我们可以通过调整维数，使之等于三维或者更高维，同时只需要改动很少的部分就可以使整个系统扩展到高维。

图 5－2 所示是整个系统的架构图，其中最高层由三部分组成：第一部分是输入、输出部分；这部分用来保证输入、输出以及手写方式的一致性。所有的输入都与我们通常手写或者可读的相同。输出也一样。换句话说，我们甚至可以直接用我们通常文献或者手写的方式输入，程序计算的结果完全可读。第二部分是核心计算部分，用它来实现我们所说的计算、证明、推理。第三部分是一个桥梁，它进行从括号代数到坐标系统的转化。这个转化有两种方式，可以以括号方式进行，剔除 syzygy 的影响，但是因为包括了括号计算的部分，我们不予考虑，我们考虑的是到坐标多项式的转化。从理论上讲，这个桥

梁应该是完善的，就是既可以从括号代数、仿射括号代数到坐标多项式的转化，也可以进行逆过程的转化。但是事实上逆过程非常困难，也是我们未来的工作重点之一。

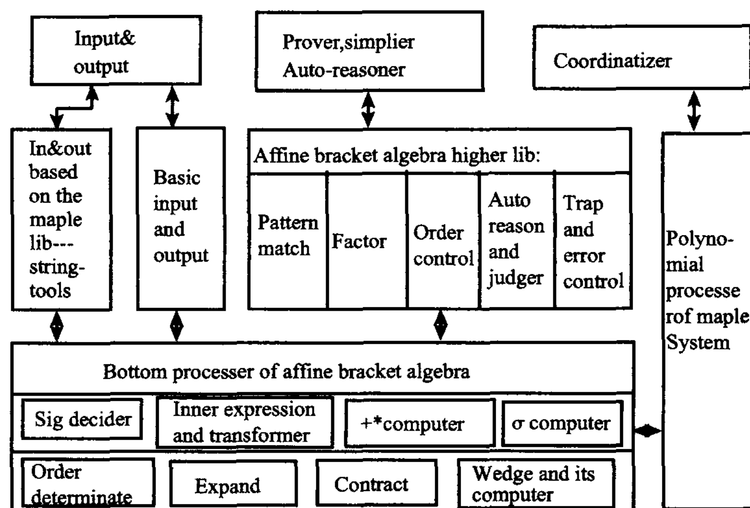


图 5-2　仿射括号代数计算系统的架构

第二个层次是利用最底层的函数来支持最高层的运算。首先我们在输入、输出部分使用了几个 Maple 的字符串函数，这部分加上自己编写的多个函数，就可以完成输入、输出任务。总体来说它们完成数字分离、括号分离、σ 分离、单项式切割、外积切割，以及它们的组合。其次是一些核心仿射括号代数的函数与算法。它们是定序判定、交换算法、收缩算法、模式库管理、模式匹配、因式分解、GCD 判定与提取、错误处理与指示等。

最底层是为了支撑整个系统而建立的各种函数和计算规则以及内部结构的控制与处理。我们用 Maple 和 C + + 编写了 67 个函数来建立整个仿射括号代数的计算规则。这些函数应该说是不友好的，它们的输入、输出都是我们定义的内部数据结构，可读性非常不好。它们与 Maple 的多项式系统是整个系统的基础部分。所有的上层命令都通过它们完成。我们可以通过输入、输出函数来访问它们。

事实上，仿射括号代数计算的输入是一个字符串结构，但是任何系统的字符串函数都不可能完成仿射括号代数计算。所以我们需要用一种好的数据结构，然后在它上面进行仿射括号的计算。这种结构应该具有如下的特性：

（1）能够毫不混淆地表达仿射括号代数的括号、边界量、外积等结构。

（2）能够区分计算用的数字和表示向量的数字。

（3）能够独立于维数，或者说可以很容易推广到更高维，扩展性好。

（4）不能与 Maple 的结构混淆，更要避免 Maple 的运算无端应用在上面。因为 Maple 作为一个编程系统还有一些问题，它不能像 C + + 一样进行各种运算与符号的重载。

我们使用的内部数据结构具有如下的形式，首先是括号结构与 σ 结构的表示：

[[维数],可计算系数,[[2 阶括号],…,[i 阶括号],
[[1 阶 σ 结构],…[i - 1 阶 σ 结构]],向量部分]

我们用下列结构表示外积 ∧ 结构：

[[维数],[2 阶 ∧ 结构],…,[i 阶 σ 结构]]

我们用下列结构表示单项式：

[[括号与 σ 部分的表示],[∧ 结构的表示]]

我们用下列结构表示多项式：[单项式 1，单项式 2，…，单项式 n]。为什么不用直接的“+”号呢？这是因为 list 结构在 Maple 里面是可计算，或者说可加的，这会导致一些 Maple 的运算强加在括号结构上，这是不需要的，也是必须避免的。

我们还注意到一点，在 Maple 10 中，list 结构并不是无限制使用的，在某些特定操作时，Maple 对 list 长度进行了限制。我们通过摸索，解决了这个问题，在所有的函数中大约有 6 个出现了这种情况。

事实上，基于 Maple 的 list 结构，效率是比较低的。因为括号代数特有的运算规则，面向对象的方法用于此系统的实现应该是很好的解决方法：

(1) 重载“*”或者说省略来定义单个括号、σ 的乘法、组成单项式。

(2) 通过重载“+”“-”等运算，实现括号单项式之间的运算，也就是组成多项式，以及进行多项式之间的加减运算。这避免了 List 结构本身的过多嵌套问题。

(3) 通过前面的整除性方法，重载“/”运算符，实现简单的除法运算，当然这种运算还是不满意的。

这样重载了四则运算符后，就可以定义括号类。括号类的成员应该包括数组列表、维数。成员函数包括：符号函数，返回括号置换后的符号；零判断，判断是否为零；拉直；syzygy

处理，自动对 syzygy 进行处理。

从括号类可以派生出括号、σ 等。基于此的 C + +实现的系统应该能在效率上大大提高，并且可重用性、可读性都大大增强。

5.3 程序的输入与输出

本书的所有符号都可以完全输入程序，并且输入输出形式一致。程序的输入输出系统也可以支持输入输出有理形式，但是通常情况下我们默认关闭这一功能，以减少判断的时间。以下输入输出例子仅是为了说明系统的功能，并不具有任何实际的背景。需要说明的是为了方便输入，程序同时支持首字母大小写的命令。

(1) 仿射括号单项式的输入和输出：在程序中，完全可以处理括号形式的字符串，但是因为 σ 的输入比较麻烦，我们还支持使用 e (1) 代替 σ (1) 来进行输入。同时我们也支持 0 到 9 共 10 个数字的输入包括它们和字母的混排，程序都可以将它们转化成内部数据结构进行计算，同样在输出时保持输入的样子。

【例 5.1】普通单项式的输入输出：

```
> myinput("3[123]e(1)"):
> myoutputall(%);
+ 3[123]σ(1)
```

```
> myinput("3[1AB]e(C)"):
> myoutputall(%);
+ 3[1AB]σ(C)
```

说明：外积的输入、输出类似。程序最主要的输入函数为 myinput（），输出函数为 myinputall（）。在没有特别指定维数参数的时候，程序默认是二维仿射平面。

（2）多项式的输入、输出：程序支持多项式的输入、输出，在定理证明和自动推理时，程序输出的每一步的多项式都是一个规则多项式。但是在其他情况，程序不限制任何形式的输入、输出，仅对维数进行控制。

【例5.2】 多项式的输入、输出：

```
> myinput("3[123][456]e(7)e(8) +
2[178][236]e(4)e(5)"):
> myoutputall(%); +
3[123][456]σ(7)σ(8) +
2[178][236]σ(4)σ(5)
```

说明：如果多项式在输入时，有同类项，程序将自动进行合并。

（3）抛物线的表示：程序支持抛物线的各种表示。

【例5.3】 抛物线的切线表示：

```
> paratangent([1,2,3,1]):
> myoutputall(%);
+ 1[13T][13T]σ(2)12 -
```

1[12T][12T]σ(3)13

说明：抛物线的表示可以用 parabola([1,2,3,X]) 来获得，表示 X 在 1、2、3、T 决定的抛物线上。不同点切线的表示可以通过调整第 4 个输入参数来获得，例如例 5.3 的输出就给出了点 1 处的切线表示。

(4) 椭圆的表示：程序支持椭圆的各种表示。

【例 5.4】 椭圆的表示：

```
> ellipse([1,2,3,X]):
> myoutputall(%); +
2[123][12O][1OX][2OX]σ(3) -
1[123][123][1OX][2OX]σ(O) -
1[12X][12X][13O][23O]σ(O) +
2[12X][1OX][13O][23O]σ(2) -
2[12X][2OX][13O][23O]σ(1)
```

说明：例 5.4 给出了 1、2、3、O 决定的椭圆，也可以通过输入参数 [1,2,3,4,5,X] 来获得由 1、2、3、4、5 点表示的椭圆。切线的相应表示也通过控制输入参数来进行。

(5) 双曲线的表示：程序支持双曲线的各种表示。

【例 5.5】 双曲线的表示：

```
> hyperbola([1,[2,3],[4,5],X]):
> myoutputall(%);
+ 1[123][145]σ(X)σ(X) -
  1[23X][45X]σ(1)σ(1)
```

说明： 例5.5给出了由两条渐近线23，45以及一点1决定的双曲线，也可以通过输入参数［1，2，3，*X*］来获得由1、2、3、*O*点表示的双曲线。切线的相应表示也通过控制输入参数来得到。

（6）程序也支持其他一些表示的输入和输出。

（7）程序证明的输出参数是可调的，可以控制输出不同的信息，同时在默认的情况下，它将给出最关键的通常也是最感兴趣的输出。对于程序中执行的错误，基于Maple的错误信息提示，程序也能给出大致的描述，并对很大一部分非法输入给出提示或者自动纠正处理。

5.4 关键运算的Maple代码

编程使用的数据结构为[数字,[[[二项括号1]、[二项括号2]]、[[三项括号1]]]、[边界量],点1数据格式必须到第二层,就是说[1,[[],[]],[[],[]]],点2]

（1）计算转换的符号，这里如果用字母，应该先要转换一下。

```
mysignal:=proc(b)
 local i,j,temp;
 temp:=0;
 for i from 1 to nops(b) do
   for j from i+1 to nops(b) do
```

```
        if b[i]>b[j] then
          temp:=temp+1;
        elif b[i]=b[j] then
          print("Error ! Wrong permutation !");
        end if;
      end do;
    end do;
    if type(temp,even) then
      return 1;
    elif type(temp,odd) then
      return -1;
    else
      print("ERROR");
    end if;
end proc:
```

(2) 进行标准格式的全排序。

```
myallorder:=proc(x)
  local num,zn,i,j,numofone,numoftwo,numofthree,
      temp,
   mytemp,numoffour; num:=nops(x);
   zn:=x;
   numofone:=nops(zn[2][1]);
   if numofone>=1 then
```

```
    for i from 1 to numofone do
      zn[1]:=zn[1]*mysignal(zn[2][1][i]);
      zn[2][1][i]:=myorder(zn[2][1][i]);
    od;
fi;
numoftwo:=nops(zn[2][2]);
if numoftwo>=1 then
    for j from 1 to numoftwo do
        zn[1]:=zn[1]*mysignal(zn[2][2][j]);
        zn[2][2][j]:=myorder(zn[2][2][j]);
    od;
fi;
numofthree:=nops(zn[3][1]);
if numofthree>=2 then
     for i from 1 to numofthree do
            for j from i+1 to numofthree do
                if zn[3][1][i][1]>zn[3][1][j]
                   [1] then
                  mytemp:=zn[3][1][j];
                  zn[3][1][j]:=zn[3][1][i];
                  zn[3][1][i]:=mytemp;
             end if;
        od;
   od;
 fi;
```

```
numoffour:=nops(zn[3][2]);
if numoffour>=1 then
  for j from 1 to numoffour do
        zn[1]:=zn[1]*mysignal(zn[3][2]
            [j]);
      zn[3][2][j]:=myorder(zn[3][2][j]);

  od;
  for i from 1 to numoffour do
      for j from i+1 to numoffour do
          if zn[3][2][i][1]>zn[3][2][j]
              [1] then
              mytemp:=zn[3][2][j];
              zn[3][2][j]:=zn[3][2][i];
              zn[3][2][i]:=mytemp;
          end if;
          if zn[3][2][i][1]=zn[3][2][j]
              [1] then
              if zn[3][2][i][2]>zn[3][2]
                  [j][2] then
                mytemp:=zn[3][2][j];
                zn[3][2][j]:=zn[3][2][i];
                zn[3][2][i]:=mytemp;
              end if;
      end if;
```

```
      od;
  od;
fi;
if numofone > =2 then
 for i from 1 to numofone do
      for j from i +1 to numofone do
           if zn[2][1][i][1] >zn[2][1][j][1]
               then
                mytemp: =zn[2][1][j];
                zn[2][1][j]: =zn[2][1][i];
                zn[2][1][i]: =mytemp;
           end if;
           if zn[2][1][i][1] =zn[2][1][j]
               [1] then
               if zn[2][1][i][2] >zn[2][1][j]
                   [2] then
                 mytemp: =zn[2][1][j];
                 zn[2][1][j]: =zn[2][1][i];
                 zn[2][1][i]: =mytemp;
               end if;
       end if;
    od;
  od;
fi;
if numoftwo > =2 then
```

```
for i from 1 to numoftwo do
  for j from i +1 to numoftwo do
      if zn[2][2][i][1] > zn[2][2][j]
         [1] then
        mytemp: =zn[2][2][j];
        zn[2][2][j]: =zn[2][2][i];
        zn[2][2][i]: =mytemp;
    end if;
    if zn[2][2][i][1] =zn[2][2][j][1]
        then
        if zn[2][2][i][2] > zn[2][2][j]
            [2] then
            mytemp: =zn[2][2][j];
            zn[2][2][j]: =zn[2][2][i];
            zn[2][2][i]: =mytemp;
        end if;
         if zn[2][2][i][2] =zn[2][2][j]
             [2]then
              if zn[2][2][i][3] >zn[2][2]
                  [j][3] then
                mytemp: =zn[2][2][j];
                zn[2][2][j]: =zn[2][2][i];
                zn[2][2][i]: =mytemp;
              end if;
          end if;
```

```
            end if;
        od;
      od;
    fi;
    if num -3 =2 then
        if zn[4] >zn[5] then
            temp: =zn[5];
            zn[5]: =zn[4];
            zn[4]: =temp;
        end if;
    end if;
    return zn;
    end proc:
```

(3)把内部的数据结构作为可读语言输出。

```
myoutput1: =proc (x)
   local i, numone, numtwo, numthree, numfour, re-
      sultone, resulttwo, resultthree, resultfour,
      resultfive, result,
      num;
    result: ="";
     numone: =nops (x [2] [1]);
     numtwo: =nops (x [2] [2]);
     numthree: =nops (x [3] [1]);
```

```
      numfour: =nops (x [3] [2]);
    if numone =0 then
       if numtwo =0 then
          if numthree =0 then
             if numfour =0 then
                if x [1] =1 then
                   if nops (x) =4 then
                      return asc (x [4]);
                   elif nops (x) =5 then
                      return cat(asc(x [4]), asc
                         (x [5]));
                   else
                      return -1;
                   fi;
                elif x [1] = -1 then
                  if nops (x) =4 then
                      return cat( "-", asc(x
                         [4]));
                   elif nops (x) =5 then
                      return cat( "-", asc(x [4]),
                         asc(x [5]));
                   else
                      return -1;
                   fi;
             else
```

```
                    return -1;
                fi;
            fi;
        fi;
      fi;
 fi;
resultone: ="";
for i from 1 to numone do
    resultone: =cat(resultone,"[", asc(x
        [2][1] [i] [1]),
    asc (x [2] [1] [i] [2], string),"]");
od;
resulttwo: ="";
for i from 1 to numtwo do
    resulttwo: =cat(resulttwo,"[", asc(x
        [2] [2] [i] [1]),
    asc(x [2] [2] [i] [2]), asc(x [2] [2]
        [i] [3]),"]");
od;
resultthree: ="";
for i from 1 to numthree do
    resultthree: =cat(resultthree," { | cc { |
        char38} k} (",
    asc(x [3] [1] [i] [1]),")");
od;
```

```
    resultfour: = "";
    for i from 1 to numfour do
        resultfour: =cat (resultfour,"{\cc {\char
            38} R} (", asc (x [3] [2] [i] [1]), asc(x
            [3] [2] [i] [2]),")");
    od;
    num: =nops (x);
    resultfive: = "";
    for i from 1 to num -3 do
       resultfive: =cat(resultfive, asc(x [3 +i]));
    od;
    if x [1]  >0 then
    return cat( "+", asc(x [1]), resultone, result-
        two, resultthree, resultfour, resultfive);
    else
    return cat(asc(x[1]), resultone, resulttwo,
        resultthree, resultfour, resultfive);
    fi;
end proc:
```

(4) 有多个 delta（12）这样的展开，x 中含有多个 e（12）这样的形式，返回一个 list；因为未必是针对整个一步，所以没有剔除公因子的操作。

```
selfexpand: =proc(x)
```

```
  local i,j,temp,result,num;
 if nops(x[3][2]) =1 then
 return selfexpand1(x);
 fi;
 if nops(x[3][2]) > =4 then
 return -1
 fi;
 if nops(x[3][2]) =3 then
 return [0];
 fi;
 temp: =[x];
 num: =1;
 for i from 1 to nops(x[3][2]) do
      result: =[];
      num: =nops(temp);
               for j from 1 to num do
                   result:=[op(result),op
                        (selfex pand1(temp[j]))];
               od;
            temp: =result;
 od;
 return result;
end proc:
```

（5）对带有外积的单项式进行展开，展开的方向由 y 决

定，确保 y 作为单点出现。

```
wedgeexpand:=proc(x,y)
    local i,sig,mark,positionx,positiony,time,
        resultone;
    sig:=0;
    mark:=0;
    positionx:=0;
    positiony:=0;
    time:=0;
    for i from 1 to nops(x[2][1]) do
        if x[2][1][i]=y then
            sig:=1;
            mark:=1;
            time:=time+1;
            positionx:=i;
        fi;
    od;
    for i from 1 to nops(x[2][2]) do
        if x[2][2][i]=y then
            sig:=1;
            mark:=2;
            time:=time+1;
            positiony:=i;
        fi;
```

```
    od;
    if sig=0 then
    return -1;
    fi;
    resultone:=[];
    if time>2 then
    return 0;
    fi;
end proc:
```

(6) 进行外积运算，返回按顺序展开的形式，在调用的时候应该用输入参数的顺序来获得好的展开，输入是2个list，或者3个list。

```
wedge:=proc()
    local resultone,resulttwo;
    if nargs=1 then
    return -1;
    fi;
    if nargs>3 then
    return -1;
    fi;
    if nops(args[1])<>2 then
    return -1;
    fi;
```

```
if nops(args[2]) < >2 then
return -1;
fi;
if nargs =3 then
   if nops(args[3]) < >2 then
   return -1;
   fi;
fi;
if nargs =2 then
   resultone: =[1,[[],[[args[1][1],args[1]
        [2],args[2][1]]]],[[],[]],args[2]
        [2]];
   resulttwo: =[ -1,[[],[[args[1][1],args
        [1][2],args[2][2]]]],[[],[]],args
        [2][1]];
fi;
if nargs =3 then
   resultone: =[1,[[],[[args[1][1],args[1][2],
    args [2][1]],[args[2][2],args[3][1],
         args[3][2]]]],[[],[]]];
   resulttwo: =[ -1,[[],[[args[1][1],args
        [1][2],args[2][2]],[args[2][1],args
        [3][1],args[3][2]]]],[[],[]]];
fi;
return [singlesim(resultone),singlesim
```

```
        (resulttwo)];
end proc;
```

（7）提取公因子，输入为一个list，list内元素为内部数据结构，但是程序并不抽取二向量，输出为剔除公因子之后的表达式，输入一个参数返回去掉公因子后的list，本函数也对整数进行gcd运算。

```
mygcd:=proc()
   local len,i,j,k,sig,s,intlist,gcdresult,
   mygcdresult,first,zn,x,num,mygcdresultone,
   mygcdresulttwo,mygcdresultthree,
        mygcdresult four;
     len:=nops(args[1]);
        x:=args[1];
        mygcdresultone:=[];
        mygcdresulttwo:=[];
        mygcdresultthree:=[];
        mygcdresultfour:=[];
for i from 1 to len do
  if nops(args[1][i][2][1])=0 then
  break;
  fi;
   for k from 1 to nops(args[1][i][2][1]) do
        first:=args[1][i][2][1][k];
```

```
            sig:=0;
            for j from 1 to len do
               if myin(first,x[j])=1 then
                  sig:=sig+1;
               fi;
            od;
            if sig=len then
               mygcdresultone:=[op
                   (mygcdresultone),first];
                 for s from 1 to len do
                   x[s]:=myremove1(first,x[s]);
                 od;
            fi;
      od;
 od;
for i from 1 to len do
   if nops(args[1][i][2][2])=0 then
  break;
  fi;
      for k from 1 to nops(args[1][i][2][2]) do
           first:=args[1][i][2][2][k];
              sig:=0;
              for j from 1 to len do
                 if myin(first,x[j])=1 then
                    sig:=sig+1;
```

```
                fi;
            od;
            if sig = len then
                mygcdresulttwo: = [op(mygcdre-
                    sulttwo),first];
                for s from 1 to len do
                    x[s]: = myremove1(first,x
                        [s]);
                od;
            fi;
    od;
od;
for i from 1 to len do

    for k from 1 to nops(args[1][i][3][1]) do
        first: = args[1][i][3][1][k];
            sig: = 0;
            for j from 1 to len do
                if myin([1,[[ ],[ ]],[[first],
                    [ ]]],x[j]) = 1 then
                    sig: = sig + 1;
                fi;
            od;
            if sig = len then
                mygcdresultthree: = [op
```

```
                    (mygcd resultthree),first];
                  for s from 1 to len do
                    x[s]: = myremove1([1,[[ ],
                         [ ]],[[first],[ ]]],x
                         [s]);
                  od;
             fi;
        od;
    od;
    for i from 1 to len do

        for k from 1 to nops(args[1][i][3][2]) do
            first: =args[1][i][3][2][k];
              sig: =0;
              for j from 1 to len do
                  if myin([1,[[ ],[ ]],[[ ],[fir-
                     st]]],x[j]) =1 then
                   sig: =sig +1;
                fi;
             od;
             if sig = len then
               mygcdresultfour: =[op
                  (mygcdresultfour),first];
                 for s from 1 to len do
                   x[s]: =myremove1([1,[[ ],[ ]
```

```
                                        ],[[ ],[ first ]]], x
                                        [s]);
                              od;
                        fi;
                od;
            od;
        intlist:=[];
            for i from 1 to len do
                intlist:=[op(intlist),x[i][1]];
            od;
            gcdresult:=igcd(op(intlist));
            for i from 1 to len do
                    x[i][1]:= x[i][1]/gcdresult;
            od;
            if nargs =1 then
                return x;
            else
                return [gcdresult,[mygcdresultone,
                mygcdresulttwo],[mygcdresultthree,
                mygcdresultfour]];
            fi;
            end proc:
```

限于篇幅，其他详细程序代码不在此列出。

5.5 例子与说明

这里的例子包括二维的仿射 incidence 部分、三维 incidence 部分、二维的二次曲线部分。关于本节的例子我们作如下说明。

(1) 在每个题目的表述部分，我们没有改变的点的字体，可以把它们看做是仿射几何命题的一般表述。

(2) 既然是定理机器证明，因此我们测试的例子，都应该由程序完成。事实上，程序完全可以完成所有的例子，但是有部分的例子，机器证明的复杂程度大于手算，这是不可避免的，因为手算的时候，会利用一些程序所不能包括的东西，使用一些没有编制到或者根本没有办法编制到程序的方法，例如例 5.4.14 给出了手算的过程，这比用程序计算要简单。当然大部分例子是机器可以轻而易举地证明，但是手算或者其他方法要耗费非常大的精力，在一些极端的情况下，似乎手算根本没有能够算出结果的可能性（指计算复杂度非常巨大)。

(3) 以下的例子中，除了例 5.4.7，其他的输出为了适应 Latex 格式，都做了格式化处理。而例 5.4.7 的文字提示是可以改变的，输出的内容也是可以改变的，这里给出的是缺省情况的输出。

(4) 以下例子中，可能有部分例子与程序的真正执行的结果不同，这可能受到输入以及程序改动的影响。而在例

5.4.14 中，程序的输出完全不同，因为这里我们给出的是一个手算的结果，程序证明过程类似于例 5.4.15、例 5.4.16。例 5.4.12 中，根据输入不同，程序输出的复杂程度也不同，为了减少篇幅，此例进行了简化处理。

（5）基于三角化的消元方法作为一种保证完整性的手段必不可少，在非常特殊的例子下，似乎仅能通过三角化，甚至坐标化方法进行消元。

（6）一些新的方法、思想还需要不断补充到程序中。

5.5.1　incidence 例子

【例 5.6】 如图 5－3 所示，过三角形 123 的点 3 引一条直线，在直线上任取一点 4，过点 1 作 24 的平行线，并与 43 交于点 5，过点 5 作 32 的平行线交 12 于点 6，则 64 平行于 31。

图 5－3　Incidence（1）

构造：自由点为 1，2，3，4。

点 5：在直线 43 上并且 15//24，可以表示为：$1\sigma(24) \wedge 43$。

点 6：在直线 12 上并且 56//32，可以表示为：$5\sigma(32) \wedge 12$。

结论：64//13，可以表示为：$[64\sigma(13)]=0$。

证明过程：

$$
\begin{aligned}
[46\sigma(31)] &= 4\sigma(31) \wedge 5\sigma(32) \wedge 12 \\
&= [125][123]\sigma(3)\sigma(4) + \\
&\quad [123][45\sigma(13)]\sigma(2) \\
&= [123]\sigma(3)\sigma(4)(12 \wedge 1\sigma(24) \wedge 43) - \\
&\quad [123]\sigma(2)(4\sigma(13) \wedge 1\sigma(24) \wedge 43) \\
&= [123][124][134]\sigma(2)\sigma(3)\sigma(4) - \\
&\quad [123][124][134]\sigma(2)\sigma(3)\sigma(4) \\
&= 0
\end{aligned}
$$

消元注释(不要展开,如果展开的话,将会有 4 项)：

$$6 := 5\sigma(32) \wedge 12$$

$$5 := 1\sigma(24) \wedge 43$$

【例 5.7】如图 5－4 所示,这是 Desargues 定理的三维情形：三维仿射空间的两个三角形,如果对应顶点的连线平行,则对应边的交点共线。

构造：

自由点:1,2,3,4。

半自由点:5 满足 25//14。

半自由点:6 满足 36//14。

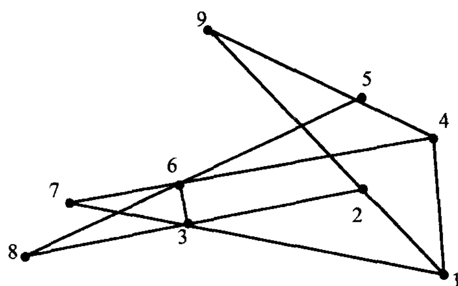


图 5 - 4　Incidence(2)

点 7:直线 13 与 46 的交点。

点 8:直线 56 与 23 的交点。

点 9:直线 45 与 12 的交点。

结论:7,8,9 三点共线。

证明过程:

$$
\begin{aligned}
[789] &= [237][569] - [239][567] \\
&= [123][436][569] - [136][456][239] \\
&= [123][456]([125][436] - [136][425]) \\
&= [125]\sigma(2) - [425]\sigma(1) \\
&= 0
\end{aligned}
$$

消元注释:

$$[237] = [123][436]$$

$$[567] = [136][456]$$

$$[569] = [125][456]$$

$$[239] = [123][425]$$

$$[136]\sigma(2) = [436]\sigma(1)$$

$$[125]\sigma(2) = [425]\sigma(1)$$

【例 5.8】 如图 5－5 所示，三角形 123 以及各边的中点 4、5、6，过点 6 且平行于 25 的直线与过点 5 且平行于 12 的直线相交于点 7，则 37 平行于 14。

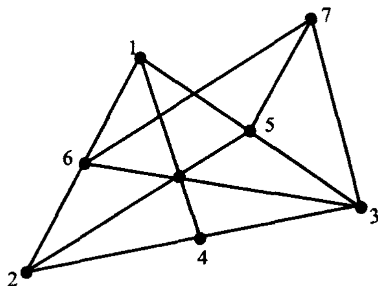


图 5－5　Incidence (3)

构造：

自由点：1，2，3。

点 4：直线 23 的中点，可以表示为：$3\sigma(2)+2\sigma(3)$。

点 5：直线 13 的中点，可以表示为：$1\sigma(3)+3\sigma(1)$。

点 6：直线 12 的中点，可以表示为：$2\sigma(1)+1\sigma(2)$。

点 7：通过点 6 且平行于 25 的直线与通过点 5 且平行于 12 的直线的交点，可以表示为：$6\sigma(25)\wedge 5\sigma(12)$。

结论：73//14，可以表示为：$[73\sigma(14)]=0$。

证明过程：

消去点 7 后的结果：

$$
\begin{aligned}
&+1[135][156]\sigma(2)\sigma(2)\sigma(4)- \\
&1[135][256]\sigma(1)\sigma(2)\sigma(4)- \\
&1[126][135]\sigma(2)\sigma(4)\sigma(5)- \\
&1[123][256]\sigma(1)\sigma(4)\sigma(5)+ \\
&1[156][345]\sigma(1)\sigma(2)\sigma(2)- \\
&1[256][345]\sigma(1)\sigma(1)\sigma(2)- \\
&1[134][256]\sigma(1)\sigma(2)\sigma(5)- \\
&1[126][345]\sigma(1)\sigma(2)\sigma(5)+ \\
&1[234][256]\sigma(1)\sigma(1)\sigma(5)
\end{aligned}
$$

消去点 6 后的结果：

$$
\begin{aligned}
&-1[125][135]\sigma(2)\sigma(3)\sigma(4)- \\
&1[135][135]\sigma(2)\sigma(2)\sigma(4)+ \\
&1[135][235]\sigma(1)\sigma(2)\sigma(4)- \\
&1[123][135]\sigma(2)\sigma(4)\sigma(5)+ \\
&1[123][235]\sigma(1)\sigma(4)\sigma(5)- \\
&1[125][345]\sigma(1)\sigma(2)\sigma(3)- \\
&1[135][345]\sigma(1)\sigma(2)\sigma(2)+ \\
&1[235][345]\sigma(1)\sigma(1)\sigma(2)+ \\
&1[134][235]\sigma(1)\sigma(2)\sigma(5)- \\
&1[123][345]\sigma(1)\sigma(2)\sigma(5)- \\
&1[234][235]\sigma(1)\sigma(1)\sigma(5)
\end{aligned}
$$

消去点 5 并去除公因子后的结果：

$$
+1[123]\sigma(4)+1[134]\sigma(2)-1[124]\sigma(3)
$$

$$= [234]\sigma(1)$$

消去点 4 的结果为零。

【例 5.9】 如图 5－6 所示，1234 是一个平行四边形，其中心为点 5，点 6、7、8、9 分别是直线 15、25、35、45 的中点。点 A 是46 和 37 的交点，点 B 是 17 和 48 的交点，点 C 是 19 和 28 的交点，点 D 是 26 和 39 的交点。则 $ABCD$ 也构成平行四边形。

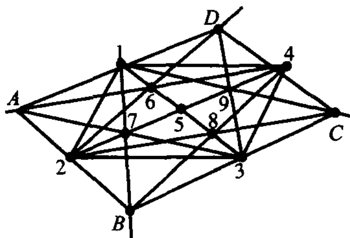


图 5－6 Incidence (4)

原题目构造如下：

自由点：1，2，3。

点 4：满足 14//23 和 34//21。

点 5：13 与 24 的交点，可以表示为：$13 \wedge 24$。

点 6：15 的中点，可以表示为：$1\sigma(5)+5\sigma(1)$。

点 7：25 的中点，可以表示为：$2\sigma(5)+5\sigma(2)$。

点 8：35 的中点，可以表示为：$3\sigma(5)+5\sigma(3)$。

点 9：45 的中点，可以表示为：$4\sigma(5)+5\sigma(4)$。

点 A：46 与 37 的交点。

点 B：17 与 48 的交点。

点 C：19 与 28 的交点。

点 D：26 与 39 的交点。

结论：$AB//DC, AD//BC$。

我们想看到它最终的依赖，所以我们去掉点 4 的构造，把点 4 转化成自由点。构造如下：

自由点：1，2，3，4。

点 5：13 与 24 的交点，可以表示为：$13 \wedge 24$。

点 6：15 的中点，可以表示为：$1\sigma(5)+5\sigma(1)$。

点 7：25 的中点，可以表示为：$2\sigma(5)+5\sigma(2)$。

点 8：35 的中点，可以表示为：$3\sigma(5)+5\sigma(3)$。

点 9：45 的中点，可以表示为：$4\sigma(5)+5\sigma(4)$。

点 A：46 与 37 的交点。

点 B：17 与 48 的交点。

点 C：19 与 28 的交点。

点 D：26 与 39 的交点。

结论：$AB//DC, AD//BC$。

推理过程（两个结论类似，这里给出一个结论的证明过程）：

程序没有给出新的消元顺序，认为按照上述构造（输入顺序）即可，说明调整顺序并不能让证明步骤简化。

消去点 D：

$$
\begin{aligned}
&+1[239][6AC]\sigma(B)+\\
&1[2AC][369]\sigma(B)-\\
&1[239][6BC]\sigma(A)-\\
&1[2BC][369]\sigma(A)
\end{aligned}
$$

消去点 C:

$$+1[129][239][68A]\sigma(B)+$$
$$1[189][239][26A]\sigma(B)+$$
$$1[129][28A][369]\sigma(B)-$$
$$1[129][239][68B]\sigma(A)-$$
$$1[189][239][26B]\sigma(A)-$$
$$1[129][28B][369]\sigma(A)$$

消去点 B:

$$-1[129][147][239][68A]\sigma(8)-$$
$$1[129][178][239][68A]\sigma(4)-$$
$$1[147][189][239][26A]\sigma(8)-$$
$$1[178][189][239][26A]\sigma(4)-$$
$$1[129][147][28A][369]\sigma(8)-$$
$$1[129][178][28A][369]\sigma(4)+$$
$$1[129][178][239][468]\sigma(A)+$$
$$1[147][189][239][268]\sigma(A)-$$
$$1[178][189][239][246]\sigma(A)-$$
$$1[129][178][248][369]\sigma(A)$$

消去点 A:

$$-1[129][147][239][367][468]\sigma(8)+$$
$$1[147][189][239][246][367]\sigma(8)-$$
$$1[129][147][268][347][369]\sigma(8)+$$
$$1[129][147][248][367][369]\sigma(8)-$$

$1[129][178][268][347][369]\sigma(4) -$

$1[129][178][239][347][468]\sigma(6) -$

$1[147][189][239][268][347]\sigma(6) +$

$1[147][189][239][268][367]\sigma(4) +$

$1[178][189][239][246][347]\sigma(6) +$

$1[129][178][248][347][369]\sigma(6)$

消去点 9:

$-1[124][147][234][367][468]\sigma(5)\sigma(5)\sigma(8) -$

$1[125][147][235][367][468]\sigma(4)\sigma(4)\sigma(8) -$

$1[147][148][234][246][367]\sigma(5)\sigma(5)\sigma(8) -$

$1[147][158][235][246][367]\sigma(4)\sigma(4)\sigma(8) +$

$1[124][147][268][346][347]\sigma(5)\sigma(5)\sigma(8) +$

$1[125][147][268][347][356]\sigma(4)\sigma(4)\sigma(8) -$

$1[124][147][248][346][367]\sigma(5)\sigma(5)\sigma(8) -$

$1[125][147][248][356][367]\sigma(4)\sigma(4)\sigma(8) +$

$1[124][178][268][346][347]\sigma(4)\sigma(5)\sigma(5) +$

$1[125][178][268][347][356]\sigma(4)\sigma(4)\sigma(4) -$

$1[124][178][234][347][468]\sigma(5)\sigma(5)\sigma(6) -$

$1[125][178][235][347][468]\sigma(4)\sigma(4)\sigma(6) +$

$1[147][148][234][268][347]\sigma(5)\sigma(5)\sigma(6) +$

$1[147][158][235][268][347]\sigma(4)\sigma(4)\sigma(6) -$

$1[147][148][234][268][367]\sigma(4)\sigma(5)\sigma(5) -$

$1[147][158][235][268][367]\sigma(4)\sigma(4)\sigma(4) -$

$1[148][178][234][246][347]\sigma(5)\sigma(5)\sigma(6) -$

$1[158][178][235][246][347]\sigma(4)\sigma(4)\sigma(6)$ -

$1[124][178][248][346][347]\sigma(5)\sigma(5)\sigma(6)$ -

$1[125][178][248][347][356]\sigma(4)\sigma(4)\sigma(6)$

在消去点 8，7，6，5 之后，程序给出收缩结果，一个单项式为：

$$+1\ [123][124][124][124][124][124]$$
$$[124][124][124][124][134][234]$$
$$\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)$$

如果按照原题目构造，在消去点 4 后程序给出结果为零。

5.5.2 抛物线例子

【例 5.10】 如图 5－7 所示，已知抛物线上 3 点 1、2、3 以及直径方向 T，过点 3 的直径与弦 12 交于点 4，过点 2 的直径与弦 13 交于点 5，则 45 平行于点 1 处的切线。

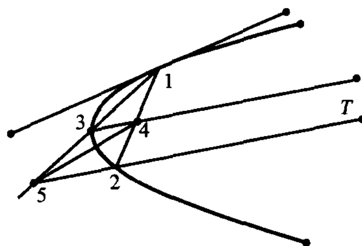


图 5－7 抛物线（1）

构造：

自由点：1，2，3，T。

点4：12与3T的交点，也就是12与过点3的直径的交点。

点5：13与2T的交点，也就是13与过点2的直径的交点。

结论：过点1的切线平行于45。

证明过程：

$$
\begin{aligned}
& [\text{tangent}(1,123T)\sigma(45)] \\
= & -[123][12T][13T]^2\sigma(2)\sigma(4)+ \\
& [124][13T]^3\sigma^2(2)- \\
& [134][12T]^2[13T]\sigma(2)\sigma(3) \\
= & [13T]\sigma(2)(-[13T]\sigma(2)+ \\
& [23T]\sigma(1)+[12T]\sigma(3)) \\
= & [13T][123]\sigma(2)\sigma(T) \\
= & 0
\end{aligned}
$$

消元注释：

$$
\begin{aligned}
\text{点}5: &= [132]T-[13T]2 \\
\text{点}4: &= [123]T-[12T]3
\end{aligned}
$$

【例5.11】 如图5-8所示，已知抛物线上3点1、2、3以及直径方向T，过12弦的中点4的直径交过点1的切线于点5，交抛物线于点6，则点6是45的中点。

构造：

自由点：1，2，3，T。

点4：弦12的中点。点5：点1处的切线与过点4的直径的交点。

点6：过点4的直径与抛物线相交的点。

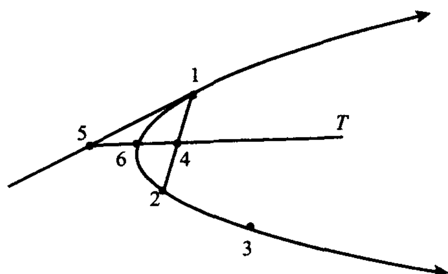


图 5-8 抛物线（2）

结论：点 6 平分线段 45。三角形 156 面积等于三角形 164 面积。

证明过程：

$$
\begin{aligned}
&[156]\sigma(4)+[146]\sigma(5)\\
&=[123][12T][24T][14T]\sigma(1)\sigma(3)\sigma(5)-\\
&\quad[123][12T][13T][14T]^3[23T]\sigma(2)\sigma(3)\sigma(4)\\
&=[123][12T][14T]\sigma(3)([24T]\sigma(1)\sigma(5)-\\
&\quad[13T][14T]^2[23T](\sigma(2)\sigma(4))\\
&=[123][12T]^2[14T]\sigma(1)\sigma(2)\sigma(3)(\sigma(5)-\\
&\quad[13T][12T][23T]\sigma(1)\sigma(4))\\
&=[123][12T]^2[14T]\sigma(1)\sigma(2)\sigma(3)([13T]^2[12T]\\
&\quad\sigma(2)\sigma(4)-[12T]^2[13T]\sigma(3)\sigma(4)-\\
&\quad[13T][12T][23T]\sigma(1)\sigma(4))\\
&=[123][12T]^3[13T][14T]\sigma(1)\sigma(2)\sigma(3)\sigma(4)\\
&\quad([13T]\sigma(2)-[12T]\sigma(3)-[23T]\sigma(1))\\
&=-[123]^2[12T]^3[13T][14T]\sigma(1)\sigma(2)\sigma(3)\sigma(4)\sigma(T)
\end{aligned}
$$

$=0$

消元注释：

$[146]=[123][12T][14T][24T]\sigma(1)\sigma(3)$

$[156]=-[123][12T][13T][14T]^3[23T]\sigma(2)\sigma(3)$

$\sigma(5)=-[13T]^2[12T]\sigma(2)\sigma(4)+$

$\qquad [12T]^2[13T]\sigma(3)\sigma(4)$

$4=\sigma(2)1+\sigma(1)2$

【例 5.12】 如图 5－9 所示，抛物线上两点的切线相交于点 4，并且过此交点的直径与弦 12 相交于点 5，则点 5 是弦 12 的中点。

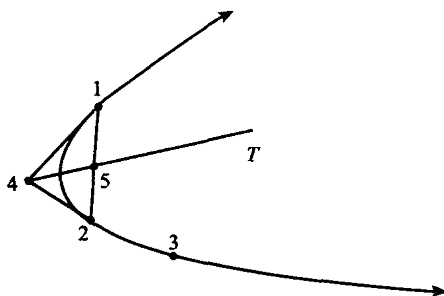


图 5－9 抛物线（3）

构造：

自由点：1，2，3，T。

点 4：点 1 切线与点 2 切线的交点。

点 5：$4T$ 与 12 的交点。

结论：点 5 平分 12。

以下为计算机证明过程：计算 $[15]\sigma(2)+[25]\sigma(1)$

" Now, Program is loaded for this prove, the followings are the result:"

" Begin……the order is not changed…… "

" Now we will remove:", 5

" The result is:"," +1 $[14T]$ σ (2) +1 $[24T]$ σ (1)"

" Now we will remove:", 4

" The result is:"," + 1$[13T][13T]\sigma(2)\sigma(2)$ - 1$[12T][13T]\sigma(2)\sigma(3)$ - 1$[23T][23T]\sigma(1)\sigma(1)$ - 1$[12T][23T]\sigma(1)\sigma(3)$"

"we use Syzygy for the first item and get :",

"1$[13T][23T]\sigma(2)\sigma(1)$ - 1$[23T][23T]\sigma(1)\sigma(1)$ - 1$[12T][23T]\sigma(1)\sigma(3)$"

"we get the gcd","$[23T]\sigma(1)$"

" Remove the gcd and we get the result:",

"1$[13T]\sigma(2)$ -1$[23T]\sigma(1)$ -1$[12T]\sigma(3)$"

" we use Syzygy and get the result", 0

注释：在此因为涉及两条切线的交点问题，实际上此交点在手算时可以直接给出，那就是弦关于二次曲线的极点。

【例 5.13】 如图 5 - 10 所示，已知抛物线上 3 点 1、2、3 以及直径方向 T，点 2 切线与点 3 切线交于点 4，12 交过点 3 的直径于点 5，则 45 平行于过点 1 的切线。

构造：

自由点：1，2，3，T。

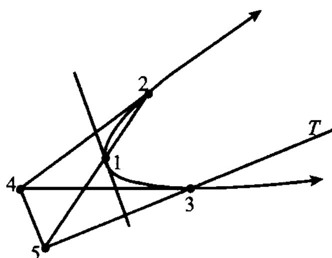


图 5－10　抛物线 (4)

点 4：点 2 处切线与点 3 处切线的交点，paratangent($123T$, 2) ∧ paratangent($123T$,3)。

点 5：弦 21 与过点 3 的直径的交点，$21\wedge 3T$。

结论：45 平行于点 1 处的切线，45//paratangent(123 T，1)。

证明过程：

$$
\begin{aligned}
&[\text{paratangent}(123T,1)\sigma(45)]\\
&=[123][12T][23T]^2\sigma(1)\sigma(3)([123][12T]^2[13T]^2\\
&\quad\sigma(2)\sigma(3)-[123][12T]^3[13T]\sigma^2(3)-\\
&\quad[123][12T]^2[13T][23T]\sigma(1)\sigma(3))\\
&=[123]^2[12T]^3[13T][23T]^2\sigma(1)\sigma^2(3)([13T]\sigma(2)-\\
&\quad[12T]\sigma(3)-[23T]\sigma(1))\\
&=-[123]^3[12T]^3[13T][23T]^2\sigma(1)\sigma^2(3)\sigma(T)\\
&=0
\end{aligned}
$$

消元注释：

$4:[123][23T]^2\sigma(1)(-[13T]^2\sigma^2(2)2+[23T]^2\sigma(1)1-$

$[12T]^2\sigma(3)3$

$5: -[23T]1+[13T]2$

$45:[12T]\sigma(3)(-[13T][23T]12+[123][12T]3T)$

$\sigma(45): -[13T][23T]\sigma(12)+[123][12T]\sigma(3)T$

5.5.3 椭圆例子

【例 5.14】 如图 5－11 所示，共轭直径平分与此直径平行的弦。

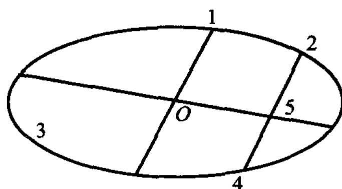


图 5－11 椭圆 (1)

构造：

自由点：1，2，3，O。

点 4：点 4 在椭圆上并且 $24//1O$。

点 5：24 与 $O1$ 的共轭半径的交点。

结论：$[25]/[54]=1$。

证明过程：

$$[25]\sigma(4)+[45]\sigma(2)$$

$$=-[24]\sigma^2(2)[12O][13O]^2\sigma(4)+$$

$$[24]\sigma^2(3)[12O]^3\sigma(4)-$$

$$
\begin{aligned}
&[24]\sigma^2(1)[12O][23O]^2\sigma(4)-\\
&[24]\sigma^3(2)[13O]^2[14O]+\\
&[24]\sigma^2(3)\sigma(2)[12O]^2[14O]+\\
&[24]\sigma^2(1)\sigma(2)[13O][23O][24O]+\\
&[24]\sigma^2(1)\sigma(2)[12O][23O][34O]\\
=&(-\sigma^2(1)[23O]^2-\sigma^2(2)[13O]^2+\\
&\sigma^2(3)[12O]^2)\sigma(O)[124]+\\
&2(-\sigma^2(1)[23O]^2-\sigma^2(2)[13O]^2+\\
&\sigma^2(3)[12O]^2)\sigma(2)[14O]+\\
&(-\sigma^2(1)[23O]^2+\sigma^2(2)[13O]^2-\\
&\sigma^2(3)[12O]^2+\\
&2\sigma(1)\sigma(2)[13O][23O])\sigma(1)[24O]\\
=&0
\end{aligned}
$$

消元注释：

$$
\begin{aligned}
[25]=&(-\sigma^2(1)[23O]^2-\sigma^2(2)[13O]^2+\\
&\sigma^2(3)[12O]^2)[12O][24]\\
[45]=&\sigma^2(1)[12O][23O][34O][24]+\\
&\sigma^2(1)[13O][23O][24O][24]-\\
&\sigma^2(2)[13O]^2[14O][24]+\\
&\sigma^2(3)[12O]^2[14O][24][12O][24]\\
&=\sigma(O)[124]+\sigma(2)[14O]-\\
&\sigma(1)[24O][12O][34O]\\
=&[14O][23O]-[13O][24O]
\end{aligned}
$$

消去点4所用的两个条件（C1，C2）：

$$C1:\sigma(O)[124]-\sigma(1)[24O]=0$$

$$C2:\sigma(1)\sigma(O)[13O][23O][124]$$
$$-\sigma^2(1)[23O]^2[14O]-$$
$$\sigma^2(2)[13O]^2[14O]+$$
$$\sigma^2(3)[12O]^2[14O]$$

【例 5.15】 如图 5－12 所示，过椭圆上点 1、4 两条平行切线与过点 2 的第 3 条切线分别交于点 5、6，16 和 45 交于点 7，则 27 平行于此两条平行切线。

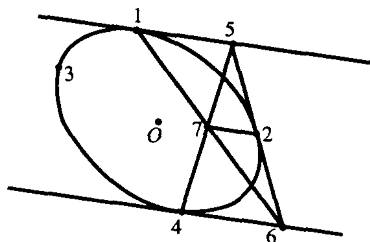


图 5－12 椭圆（2）

构造

自由点：1，2，3，O。

点 4：点 4 的切线平行于点 1 的切线。

点 5：点 1 的切线与点 2 的切线的交点。

点 6：点 2 的切线与点 4 的切线的交点。

点 7：16 与 54 的交点。

结论：27//51，27//64。

证明过程（两个结论证明过程类似，此处给出一个结论的

证明过程)：

计算 $[271]\sigma(5)-[275]\sigma(1)$，程序改变消元顺序为 [7, 6,4,5]

消去点 7：

$$+1[126][145]\sigma(5)+1[145][256]\sigma(1)-1[125][456]\sigma(1)$$

消去点 6：

$$-2[123][12O][12O][145][245]\sigma(1)\sigma(3)\sigma(5)+$$
$$1[123][123][12O][145][245]\sigma(1)\sigma(5)\sigma(O)+$$
$$2[123][124][12O][12O][145]\sigma(3)\sigma(5)\sigma(5)-$$
$$1[123][123][124][12O][145]\sigma(5)\sigma(5)\sigma(O)+$$
$$2[12O][12O][145][235][245]\sigma(1)\sigma(1)\sigma(3)-$$
$$1[123][12O][145][235][245]\sigma(1)\sigma(1)\sigma(O)-$$
$$2[124][12O][12O][145][235]\sigma(1)\sigma(3)\sigma(5)+$$
$$1[123][124][12O][145][235]\sigma(1)\sigma(5)\sigma(O)+$$
$$2[125][145][23O][23O][245]\sigma(1)\sigma(1)\sigma(1)-$$
$$1[123][125][145][23O][245]\sigma(1)\sigma(1)\sigma(O)-$$
$$2[124][125][12O][12O][345]\sigma(1)\sigma(3)\sigma(5)+$$
$$2[125][12O][12O][134][245]\sigma(1)\sigma(3)\sigma(5)+$$
$$1[123][124][125][12O][345]\sigma(1)\sigma(5)\sigma(O)-$$
$$1[123][125][12O][134][245]\sigma(1)\sigma(5)\sigma(O)$$

消去点 4：

$$-2[123][12O][12O][15O][25O]\sigma(1)\sigma(3)\sigma(5)+$$

$$1[123][123][12O][15O][25O]\sigma(1)\sigma(5)\sigma(O)-$$
$$2[123][12O][12O][12O][15O]\sigma(3)\sigma(5)\sigma(5)+$$
$$1[123][123][12O][12O][15O]\sigma(5)\sigma(5)\sigma(O)+$$
$$2[12O][12O][15O][235][25O]\sigma(1)\sigma(1)\sigma(3)-$$
$$1[123][12O][15O][235][25O]\sigma(1)\sigma(1)\sigma(O)+$$
$$2[12O][12O][12O][15O][235]\sigma(1)\sigma(3)\sigma(5)-$$
$$1[123][12O][12O][15O][235]\sigma(1)\sigma(5)\sigma(O)+$$
$$2[125][15O][23O][23O][25O]\sigma(1)\sigma(1)\sigma(1)-$$
$$1[123][125][15O][23O][25O]\sigma(1)\sigma(1)\sigma(O)+$$
$$2[125][12O][12O][12O][35O]\sigma(1)\sigma(3)\sigma(5)-$$
$$2[125][12O][12O][13O][25O]\sigma(1)\sigma(3)\sigma(5)-$$
$$1[123][125][12O][12O][35O]\sigma(1)\sigma(5)\sigma(O)+$$
$$1[123][125][12O][13O][25O]\sigma(1)\sigma(5)\sigma(O)$$

消去点 5，结果为零。基本上这个过程项数过多，令人很不满意。

【例 5.16】 如图 5－13 所示，椭圆上两点 4、5 与中心 O 的连线分别平行于另外两点 1、2 的切线，点 1、2 处的切线交于点 6，46 和 56 分别交 $1O$ 和 $2O$ 于点 7、8，18 和 27 交于点 9，则点 6、9、O 共线。

构造：

自由点：1，2，3，O。

点 4：$O4$ 是一条半径，并且 $O4$ 平行于点 1 处的切线。

点 5：$O5$ 是一条半径，并且 $O5$ 平行于点 2 处的切线。

点 6：点 1 的切线与点 2 的切线的交点。

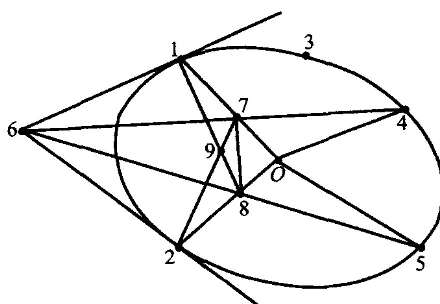


图5-13 椭圆(3)

点7：46与1O的交点。

点8：56与2O的交点。

点9：27与18的交点。

结论：点6，9，O共线。

证明过程：计算 [69O]

消去点9，结果为：

$$+1[127][68O]+1[16O][278]$$

消去点8，结果为：

$$+1[127][26O][56O]+1[16O][256][27O]$$

消去点7，结果为：

$$+1[146][26O][56O]+1[16O][256][46O]$$

消去点6，结果为：

$$-4[12O][12O][12O][12O][12O][12O][134]$$

$$[23O][35O]\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)+$$

$$4[123][123][123][12O][12O][12O][134]$$
$$[23O][35O]\sigma(3)\sigma(3)\sigma(3)\sigma(O)\sigma(O)\sigma(O)-$$
$$1[123][123][123][123][123][123][134]$$
$$[23O][35O]\sigma(O)\sigma(O)\sigma(O)\sigma(O)\sigma(O)\sigma(O)-$$
$$4[12O][12O][12O][12O][12O][12O][13O]$$
$$[235][34O]\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)\sigma(3)+$$
$$4[123][123][123][12O][12O][12O][13O]$$
$$[235][34O]\sigma(3)\sigma(3)\sigma(3)\sigma(O)\sigma(O)\sigma(O)-$$
$$1[123][123][123][123][123][123][13O][235]$$
$$[34O]\sigma(O)\sigma(O)\sigma(O)\sigma(O)\sigma(O)\sigma(O)$$

此式可以分解为如下结果：

$$-([134][23O][35O]+[13O][235][34O])$$
$$(2[12O]^3\sigma^3(3)-[123]^3\sigma^3(O))^2$$

接着对程序点 4，5，只能用三角化的方法进行消去，过程非常冗长，结果得证。

证明过程中产生的非退化条件是 $2[12O]^3\sigma^3(3)-[123]^3\sigma^3(O)\neq 0$。

【例 5.17】 如图 5 – 14 所示，椭圆的外切三角形的重心与椭圆的中心重合，如果切点分别是各边的中点。

构造：

自由点：1，2，3，4，5。

点 6：椭圆过点 1 的切线与过点 2 的切线的交点。

点 7：椭圆过点 1 的切线与过点 3 的切线的交点。

点 8：椭圆过点 2 的切线与过点 3 的切线的交点。

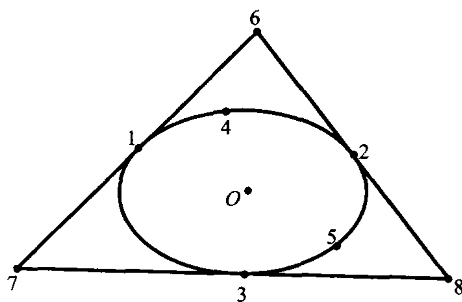


图 5－14　椭圆（4）

结论：如果点 1，2，3 分别是 67，68，78 的中点，则三角形 678 的重心重合于椭圆的中心。

证明过程：计算

$$
\begin{aligned}
&6\sigma(7)+7\sigma(6)+6\sigma(8)+8\sigma(6)+7\sigma(8)+8\sigma(7)\\
&=2z(a+b-c)+2y(a-b+c)+2x(-a+b+c)\\
&=2a(z+y-x)+2b(z+x-y)+2c(x+y-z)\\
&=[145][234][235](-[145][234][235]\sigma(1)+\\
&\quad[134][135][245]\sigma(2)+[124][125][345]\sigma(3))1+\\
&\quad[134][135][245]([145][234][235]\sigma(1)-\\
&\quad[134][135][245]\sigma(2)+[124][125][345]\sigma(3))2+\\
&\quad[124][125][345]([145][234][235]\sigma(1)+\\
&\quad[134][135][245]\sigma(2)-[124][125][345]\sigma(3))3\\
&=0
\end{aligned}
$$

消元注释：

6 = ellitangent([1,2,3,4,5,1]) ∧

ellitangent([1,2,3,4,5,2])

$=[145][234][235]1+[134][135][245]2-[124][125][345]3$

$=a+b-c$

$\sigma(6)=[145][234][235]\sigma(1)+[134][135][245]\sigma(2)-[124][125][345]\sigma(3)$

$=x+y-z$

7 = ellitangent([1,2,3,4,5,1]) ∧ ellitangent([1,2,3,4,5,3])

$=[145][234][235]1-[134][135][245]2+[124][125][345]3$

$=a-b+c$

$\sigma(7)=[145][234][235]\sigma(1)-[134][135][245]\sigma(2)+[124][125][345]\sigma(3)$

$=x-y+z$

8 = ellitangent([1,2,3,4,5,2]) ∧ ellitangent([1,2,3,4,5,3])

$=-[145][234][235]1+[134][135][245]2+[124][125][345]3$

$=-a+b+c$

$\sigma(8)=-[145][234][235]\sigma(1)+[134][135][245]\sigma(2)+[124][125][345]\sigma(3)$

$=-x+y+z$

5.5.4　双曲线例子

【例 5.18】 如图 5－15 所示，任作一条直线分别交双曲线及其渐近线于点 1、7、9、8，则点 6 平分 89，当且仅当点 6 平分 17。

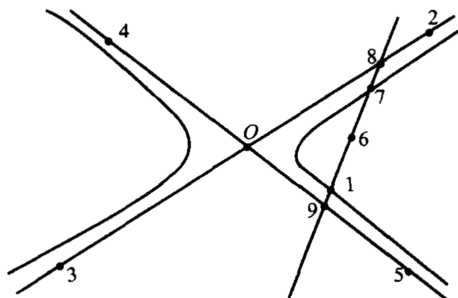


图 5－15　双曲线（1）

构造：

自由点：1，23，45。

半自由点：点 7 在双曲线 hyperbola（1，23，45）上。

半自由点：点 6 是直线 17 上的点。

点 8：直线 17 与渐近线 23 的交点。

点 9：直线 17 与渐近线 45 的交点。

结论：[61] = [67]⇔[68] = [69]。

证明过程：

$$[69]\sigma(8)+[68]\sigma(9)$$

$$=([457][61]-[145][67])\times([321]\sigma(7)-[327]\sigma(1))+$$

$$([321][67]-[327][61])\times([457]\sigma(1)-[451]\sigma(7))$$

$$=[457][321][61]\sigma(7)-[451][321][67]\sigma(7)-$$
$$[457][327][61]\sigma(1)+[451][327][67]\sigma(1)+$$
$$[321][457][67]\sigma(1)-[327][457][61]\sigma(1)-$$
$$[451][321][67]\sigma(7)+[327][451][61]\sigma(7)$$
$$=(1)+(2)+(3)+(4)+(5)+(6)+(7)+(8)$$
$$=([457][321]\sigma(7)+[145][327]\sigma(7)-2[327][457]\sigma(1))$$
$$([67]\sigma(1)+[61]\sigma(7))$$

消元注释：

$$(1)+(4)+(5)+(8)$$
$$=([457][321]+[451][327])([61]\sigma(7)+[67]\sigma(1))$$
$$(2)+(3)+(6)+(7)$$
$$=-2([451][321][67]\sigma(7)+[457][327][61]\sigma(1))$$
$$=-2\frac{[327][457]\sigma(1)}{\sigma(7)}([67]\sigma(1)+[61]\sigma(7))$$
$$7:[123][145]\sigma^2(7)-[237][457]\sigma^2(1)=0$$

【例 5.19】 如图 5－16 所示，双曲线上 3 点 1、2、3，过点 3 与一条渐近线平行的直线和过点 1 与另外一条渐近线平行的直线相交于点 7，弦 12 与第一条渐近线交于点 6，则 67 平行于弦 23。

说明： 本例可以不按照下述方法构造，机器证明过程类似于最后的两个例子，项数比较多，但是如果我们用如下的构造方式，充分利用射影的结果，可以给出如下简洁的证明。

构造：

自由点：1，2，3。

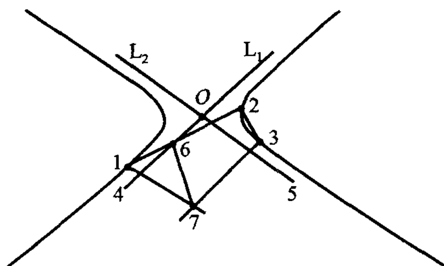


图5-16 双曲线（2）

无限远处的自由点：4，5。

极点：O：=pole（12345，45）。

点6：12与$O4$的交点，$12 \wedge O4$。

点7：34与15的交点，$34 \wedge 15$。

结论：$[67\sigma(23)]=0$。

证明过程：

$$
\begin{aligned}
[67\sigma(23)] &= [367]\sigma(2)-[267]\sigma(3)\\
&= [124][135][34O]\sigma(2)-[125][134][24O]\sigma(3)\\
&= [124][125][134][135][234]([345]\sigma(2)-\\
&\quad [245]\sigma(3))\\
&= 0
\end{aligned}
$$

消元注释：

$$
\begin{aligned}
[267] &= [2(12 \wedge O4)(34 \wedge 15)]\\
&= [125][134][24O]\\
[367] &= [3(12 \wedge O4)(34 \wedge 15)]\\
&= [124][135][34O]
\end{aligned}
$$

【例 5.20】 如图 5-17 所示，双曲线上取两点 1、6，过点 6 作一条渐近线的平行线与点 1 的切线交于点 7，过点 1 作一条渐近线的平行线与点 6 的切线交于点 8，则 78 平行于弦 16。

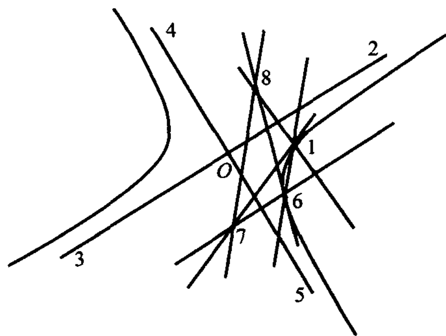


图 5-17 双曲线 (3)

构造：

自由点：1，23，45。

半自由点：6 在双曲线上。

点 7：在过点 1 的切线上，且 67 平行于渐近线 23。

点 8：在过点 6 的切线上，且 18 平行于渐近线 45。

结论：78//16。

证明过程：计算 $[167]\sigma(8)-[168]\sigma(7)$

消去点 8 后得到结果：

$$+1[123][167][346]\sigma(1)\sigma(2)\sigma(5)-$$
$$1[123][167][356]\sigma(1)\sigma(2)\sigma(4)-$$
$$1[123][167][246]\sigma(1)\sigma(3)\sigma(5)+$$

$$1[123][167][256]\sigma(1)\sigma(3)\sigma(4)+$$
$$1[136][136][145]\sigma(2)\sigma(2)\sigma(7)-$$
$$1[123][136][146]\sigma(2)\sigma(5)\sigma(7)+$$
$$1[123][136][156]\sigma(2)\sigma(4)\sigma(7)+$$
$$1[126][126][145]\sigma(3)\sigma(3)\sigma(7)-$$
$$2[126][136][145]\sigma(2)\sigma(3)\sigma(7)+$$
$$1[123][126][146]\sigma(3)\sigma(5)\sigma(7)-$$
$$1[123][126][156]\sigma(3)\sigma(4)\sigma(7)$$

消去点7并进行交换后得到：

$$+1[123][126][145][345][456]\sigma(1)\sigma(2)\sigma(3)\sigma(6)\sigma(6)$$
$$-1[126][236][345][456][456]\sigma(1)\sigma(1)\sigma(1)\sigma(2)\sigma(3)$$

提取公因子为：

$$[126][345][456]\sigma(1)\sigma(2)\sigma(3)([123][145]\sigma(6)\sigma(6)-$$
$$[236][456]\sigma(1)\sigma(1))$$
$$=0$$

消去半自由点规则为点6在双曲线上：

$$[236][456]\sigma^2(1)-[123][145]\sigma^2(6)=0$$

【例5.21】 如图5-18所示，过双曲线上两点1，6做切线，分别交渐近线于点8、9、*A*、7，则89平行于*A*7。

构造：

自由点：1，23，45。

半自由点：6在双曲线上。

点7：在点1的切线与渐近线23的交点。

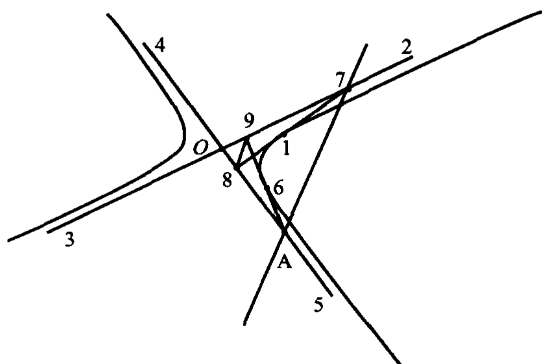


图 5-18 双曲线 (4)

点 8：在点 1 的切线与渐近线 45 的交点。

点 9：在点 6 的切线与渐近线 23 的交点。

点 A：在点 6 的切线与渐近线 45 的交点。

结论：89//A7。

证明过程（为了不至于混淆，这里我们用 A 替代上面的 0)：计算 $[78A]\sigma(9)-[79A]\sigma(8)$

消元顺序为点 9，A，8，7，6。

消去点 9 后的结果为：

$$+1[246][78A]\sigma(3)\sigma(5)-1[346][78A]\sigma(2)\sigma(5)-1[256][78A]\sigma(3)\sigma(4)+1[356][78A]\sigma(2)\sigma(4)-1[27A][456]\sigma(3)\sigma(8)+1[37A][456]\sigma(2)\sigma(8)-1[246][37A]\sigma(5)\sigma(8)+1[27A][346]\sigma(5)\sigma(8)+1[256][37A]\sigma(4)\sigma(8)-1[27A][356]\sigma(4)\sigma(8)$$

消去点 A 后的结果为：

$$
\begin{aligned}
&-1[246][246][578]\sigma(3)\sigma(3)\sigma(5)\ +\\
&1[246][256][478]\sigma(3)\sigma(3)\sigma(5)\ -\\
&1[246][356][478]\sigma(2)\sigma(3)\sigma(5)\ +\\
&1[236][246][478]\sigma(3)\sigma(5)\sigma(5)\ -\\
&1[236][246][578]\sigma(3)\sigma(4)\sigma(5)\ +\\
&2[246][346][578]\sigma(2)\sigma(3)\sigma(5)\ -\\
&1[256][346][478]\sigma(2)\sigma(3)\sigma(5)\ -\\
&1[346][346][578]\sigma(2)\sigma(2)\sigma(5)\ +\\
&1[346][356][478]\sigma(2)\sigma(2)\sigma(5)\ -\\
&1[236][346][478]\sigma(2)\sigma(5)\sigma(5)\ +\\
&1[236][346][578]\sigma(2)\sigma(4)\sigma(5)\ +\\
&1[246][256][578]\sigma(3)\sigma(3)\sigma(4)\ -\\
&1[256][256][478]\sigma(3)\sigma(3)\sigma(4)\ -\\
&1[256][346][578]\sigma(2)\sigma(3)\sigma(4)\ -\\
&1[236][256][478]\sigma(3)\sigma(4)\sigma(5)\ +\\
&1[236][256][578]\sigma(3)\sigma(4)\sigma(4)\ -\\
&1[246][356][578]\sigma(2)\sigma(3)\sigma(4)\ +\\
&2[256][356][478]\sigma(2)\sigma(3)\sigma(4)\ +\\
&1[346][356][578]\sigma(2)\sigma(2)\sigma(4)\ -\cdots\cdots
\end{aligned}
$$

消去点 8：……

消去点 7 后去除公因子并进行分解的结果为：

$$
\begin{aligned}
&\{+1[124][345]\sigma(5)-1[134][245]\sigma(5)\ -\\
&1[125][345]\sigma(4)+1[135][245]\sigma(4)\}\\
&\{[123][145]\sigma(6)\sigma(6)-[236][456]\sigma(1)\sigma(1)\}
\end{aligned}
$$

$= 0$

消元半自由点规则仍然为点 6 在双曲线上：$[236][456]\sigma^2(1) - [123][145]\sigma^2(6) = 0$。事实上，最后的结果还可以继续分解下去。

参考文献

[1] 吴文俊．数学机械化［M］．北京：科学出版社，2003.

[2] 王东明．消去法及其应用［M］．北京：科学出版社，2002.

[3] 王浩．向机械化数学前进［M］．北京：科学出版社，1959.

[4]（英）科克肖特 A，沃尔特斯 F B. 圆锥曲线的几何性质［M］．上海：上海教育出版社，2002.

[5] 王坦．协作学习——原理与策略［M］．北京：学苑出版社，2001.

[6] 衷仁保．几何定理机器证明的基本原理［M］．北京：科学出版社，1989.

[7] 莫宗坚，蓝以中，赵春来．代数学［M］．北京：北京大学出版社，1986.

[8] 冯克勤．交换代数基础［M］．北京：高等教育出版社，1985.

[9] 石赫. 数学机械化引论 [M]. 长沙：湖南教育出版社，1998.

[10] 数学手册编写组. 数学手册 [M]. 北京：高等教育出版社，1979.

[11] 王东明，等. 符号计算选讲 [M]. 北京：清华大学出版社，2003.

[12] 王东明，夏壁灿. 计算机代数 [M]. 北京：清华大学出版社，2003.

[13] Hilbert D. Uber die Theorie der algebraischen Formen. Math Annalen, 1890 (36): 473 ~ 531.

[14] Hilbert D. Theory of Algebraic Invariants [M]. Cambridge University Press, 1993.

[15] Hilbert D. Uber die vollen Invarianten systeme. Math Annalen, 1893 (42): 70 ~ 313.

[16] Sturmfels B, Whiteley W. On the Synthetic Factorization of Homogeneous Invariants. Journal of Symbolic Computation, 1991 (11): 439 ~ 454.

[17] Sturmfels B. Algorithms in Invariant Theory. New York: Springer Press, 1993.

[18] Shangching Chou, Xiaoshan Gao, Jingzhong Zhang. Automated generation of readable proofs with geometric invariants: 1, Multiple and shortest proof generation. Joural of Automated Reasoning, 1996 (17): 325 ~ 347.

[19] Shangching Chou, Xiaoshan Gao, Jingzhong Zhang. Automated generation of readable proofs with geometric invariants, 2,

Multiple and shortest proof generation. Joural of Automated Reasoning, 1996 (17): 349 ~ 370.

[20] Shangching Chou, Xiaoshan Gao, Jingzhong Zhang. Mechanical geometry theorem proving by vector calculation. Procecding of ISSAC, Kiev: ACM Press, 1993: 284 ~ 291.

[21] Shangching Chou, Xiaoshan Gao. Mechanical theorem proving in Riemannian geometry using Wu's method. Computer Mathematics, Singapore: World Scientific, 1993: 135 ~ 158.

[22] Shangching Chou, Xiaoshan Gao, Jingzhong Zhang. Automated generation of traditional proofs in solid geometry, Joural of Automated Reasoning. 1995 (14): 257 ~ 291.

[23] Dongming. Wang. Clifford Algebraic Calcaulus for geometric reasoning with application to cumputer vision, In Automated deduction in geometry. Berlin: Springer Press, 1997: 115 ~ 140.

[24] Wentsun. Wu On the decision problem on application and the mechanization of theorem-proving in elementary geometry. In: Scientia Sinica, 1978 (21): 159 ~ 172.

[25] Wentsun. Wu Mechanic Theorem proving in elementary geometry and differential geometry. In: Proc. 1980 Beijing DD-symposium, 1980 v. (2): 1073 ~ 1092.

[26] Wentsun. Wu Basic principles of mechanical theorem-proving in elementary geometry. Journal of System Science, 1984 (4): 207 ~ 235.

[27] Wentsun. Wu Basic principles of mechanical theorem-proving in geometries. Beijing: Science Press, 1984.

[28] Hongbo. Li, Nanbin Cao. Geometry decomposition and program implement based on conform geometric algebra. submitted to system sciences and mathematics.

[29] Hongbo. Li, Yihong. Wu. Automated Theorem Proving with Bracket Algebra in Projective Geometry. Computer Mathematics, World Scientific, 2000: 120 ~ 129.

[30] Hongbo. Li, Yihong. Wu. Automated Short Proof Generation for Projective Geometric Theorems with Cayley and Bracket algebras. I. Incide
nce Geometry. Journal of symbolic Computation, 2003 (36): 717 – 762.

[31] Hongbo. Li, Yihong. Wu. Automated Short Proof Generation for Projective Geometric Theorems with Cayley and Bracket algebras. Conic Geometry. Journal of symbolic computation. 2003 (36): 763 – 809.

[32] Hongbo. Li, Yihong. Wu. Automated Theorem Proving in Incidence Geometry-A Bracket Algebra Based Elimination Method. Automated Deduction in Geometry, 2000: 199 – 227.

[33] Hongbo. Li. Hyperbolic Geometry with Clifford Algebra. ACta Applicandae Mathematicae, 1997 (48): 317 ~ 358.

[34] Hestenes David, Hongbo Li, Rockwood Alyn. An algebra of planes and simplices. In: Geometric Computing with Clifford Algebra, G. Sommer, ed al. Springer Heidelberg, 2003: 3 ~ 26.

[35] Hongbo. Li, Hestenes David, Rockwood Alyn. Generalized homogeneous Coordinates for computational geometry. In: Geometric Computing with Clifford Algebra, G. Sommer, ed al. Springer

Heidelberg, 2003: 27 ~60.

[36] Hongbo. Li, Hestenes David, Rockwood Alyn. Spherical conformal geometry with geometric algebra. In: Geometric Computing with Clifford Algebra, 2002: Springer Heidelberg, 2003: 61 ~76.

[37] Hongbo. Li, David Hestenes, Alyn Rockwood. An universal model for conformal geometries of Euclidean, spherical and double-hyperbolic spaces. Geometric Computing with Clifford Algebra, Springer Heidelberg, 2003: 77 ~104.

[38] Hongbo. Li, Sommer Gerald. Coordinate-free projective geometry for computer vision. Geometric Computing with Clifford Algebra, Springer Heidelberg, 2003: 415 ~454.

[39] Hongbo. Li. Mechanical theorem proving in differential geometry, Mathematics Mechanization and Applications, Academic Press, 2004: 147 ~174.

[40] Hongbo. Li. Clifford algebra approaches to automated geometry theorem proving. Mathematics Mechanization and Applications, X. -S. Gao and D. Wang (eds.), London: Academic Press, 2004: 205 ~230.

[41] Hongbo. Li. Hyperbolic conformal geometry with Clifford algebra. International Journal of Theoretical Physics, 2001 (40): 79 ~91.

[42] Hongbo. Li. Hyperbolic geometry. Advances in Geometric Algebra with Applications in Science and Engineering, Birkhauser Boston, 2002: 64 ~88.

[43] Hongbo. Li. Automated theorem proving. Advances in

Geometric Algebra with Applications in Science and Engineering, 2002: 112 ~122.

[44] Xiaorong. Hou, Hongbo. Li, Dongming Yang, Lu Yang. "Russian killer" No. 2: a challenging geometric theorem with machine vs. human proofs. Math of Intelligencer, 2001 (23): 9 ~15.

[45] Hongbo. Li. Trifocal tensors with Grassmann-Cayley algebra. Robot Vision, Springer Berlin Heidelberg, 2001: 237 ~244.

[46] Hongbo. Li. Clifford algebraic computing in artificial vision. Mathematics Mechanization, People's Education Press, 2001: 187 ~201.

[47] Hongbo. Li. Automated geometry theorem proving in the homogeneous model with Clifford bracket algebra. Applications of Geometric Algebra in Computer Science and Engineering, Birkhauser, 2001: 69 ~78.

[48] Hongbo. Li. Clifford Algebras and Homogeneous Geometric Models. Some Problems on the Protein Structure Analysis. CCAST-WL Workshop Series 147, 2001: 91 ~121.

[49] Hongbo Li. Clifford algebra, geometric computing and reasoning. Chinese Advance in Math, 2004 (32): 405 ~415.

[50] Hongbo Li, Chen Ying. A bracket method for judging the intersection of convex bodies. Computer Mathematics, World Scientific, 2004: 227 ~239.

[51] Hongbo Li, Yihong Wu. Automated Theorem Proving in Projective Geometry with Cayley and Bracket Algebras I. Incidence Geometry. Journal of Symbolic Computation, 36 (5): 717 ~762.

[52] Hongbo Li, Yihong Wu. Automated Theorem Proving in Projective Geometry with Cayley and Bracket Algebras II. Conic Geometry. J. of Symbolic Computation, 36 (5): 763 ~ 809.

[53] Hongbo Li. Clifford Algebras and Geometric Computation. Geometric Computation, World Scientific, 2003: 221 ~ 247.

[54] Hongbo Li. Automated Geometric Theorem Proving, Clifford Bracket Algebra and Clifford Expansions. Trends in Mathematics: Advances in Analysis and Geometry, Birkhauser Basel, 2004: 345 ~ 363.

[55] Hongbo Li. Algebraic Representation, Elimination and Expansion in Automated Geometric Theorem Proving. Automated Deduction in Geometry, F. Winkler, ed al. Springer, Berlin, Heidelberg, 106 ~ 123.

[56] Hongbo Li. Symbolic Computation in the Homogeneous Geometric Model with Clifford Algebra. In: Proc. ISSAC 2004, J. Gutierrez, ed al. ACM Press, 221 ~ 228.

[57] Haiman Mark. Proof Theory for Linear Lattices. Advances in mathematics. Volume58, number3, 1985.

[58] Doran, Chris. Circle and sphere blending with conformal geometric algebra. eprint arXiv: cs/0310017.

[59] Yihong Wu. Bracket Algebra, Affine Bracket Algebra and Mechanical Theorem Proving. PhD paper 2001.

[60] Hee. Cheo. Cho, Hyeong. In. Choi, Song-Hwa. Kwon, Doo. Seok. Lee, Nam-Soo. Wee. Clifford algebra, Lorentzian geometry, and rational parametrization of canal surfaces, Computer Aided

Geometric Design, 2004 (21): 327 ~ 339.

[61] J. Richter-Gebert. Mechanical Theorem Proving in Projective Geometry. Annals of Math. and Artificial Intelligence, 1995: 159 ~ 171.

[62] J. Richter-Gebert. Realization Spaces of Polytopes. Computational Algebraic Geometry of Projective Configurations Journal of Symbolic Computation, 1991 (11): 595 ~ 618.

[63] Sturmfels B. Algorithms in Invariant Theory. New York: Springer, 1993: 106 ~ 108.

[64] Sturmfels B, Whiteley W. On the Synthetic Factorization of Homogeneous Invariants. Journal of Symbolic Computation, 1991 (11): 439 ~ 454.

[65] Dongming Wang. Elimination Methods. New York: Springer, 2001.

[66] White N. The Bracket Ring of Combinatorial Geometry I. Trans. Amer. Math. Soc. 1975 (202): 79 ~ 103.

[67] White. Multilinear Cayley Factorization. Journal of Symbolic Computation, 1991 (11): 421 ~ 438.

[68] McMillan T, White N. The Dotted Straightening Algorithm. Journal of Symbolic Computation, 1991 (11): 471 ~ 482.

[69] Whiteley W. Invariant Computations for Analytic Projective Geometry. Journal of Symbolic Computation, 1991 (11): 549 ~ 578.

[70] Wentsun Wu. Mathematics Mechanization. Science Press/Kluwer Academic, Beijing, 2000.

[71] Doublilet P, Rota G C, Stein J. On the Foundations of Combinatorial Theory IX: Combinatiorial Methods in Invariant Theory, Journal of Symbolic Computation, 1974 (57): 185 ~216.

[72] Concini C De, Procesi C. A Characteristic Free Approach to Invariant Theory. Advanced Math, 1976 (21): 330 ~354.

[73] Mainetti M, C H Yan. Geometric Identities in Lattice Theory. J. Comb. Theory, Series A, 2000 (91): 411 ~450.

[74] Mainetti M, C H Yan. Arguesian Identities in Linear Lattices. Adv. Math. 1999 (144): 50 ~93.

[75] Chris Doran, Anthony Lasenby and Joan Lasenby. Conformal Geometry, Euclidean Space and Geometric Algebra. J. Winkler, ed al. Uncertainty in Geometric Computations, Kluwer, 2002.

[76] Chris Doran. New Advances in Geometric Algebra. Uncertainty in Geometric Computations. Kluwer, 2002.

[77] Chris Doran. Circle and Sphere Blending with Conformal Geometric Algebra, Submitted to Computer Aided Geometric Design.

[78] Leo Dorst, Daniel Fontijne. An Algebraic Foundation for Object-Oriented Euclidean Geometry, Proceeding of ITM, 2003.

[79] Daniel Fontijne, Leo Dorst. Modeling 3D Euclidean Geometry-Performance and Elegance of Five Models of 3D Euclidean Geometry in a Ray Tracing Application, IEEE Trans on Computer Graphics and Applications, 2003 (23): 68 ~78.

[80] Sommer G, Geometric Computing with Clifford Algebras. Springer Press, 2001.

[81] Hestenes D. Invariant Body Kinematics II. Reaching and Neurogeometry, Neural Networks, 1994 (7): 79 ~88.

[82] Hitzer Eckhard, KamiWaAi. Interactive 3D Sketching with Java Based on Cl (4, 1) Conformal Model of Euclidean Space. Advances in Applied Clifford Algebras, 2003 (13): 11 ~45.

[83] Joan Lasenby, Sahan Gamage, Maurice Ringer. Modelling Motion: Tracking, Analysis and Inverse Kinematics. AFPAC 2000, Springer, 2000: 104 ~114.

[84] Joan Lasenby. Using Clifford/Geometric Algebra in Robotics, Proceeding of NATO Computational Noncommutative Algebra and Applications, 2003.

[85] Anthony Lasenby, Joan Lasenby. Surface Evolution and Representation Using Geometric Algebra. The Mathematics of Surfaces IX. Springer, 2000: 144 ~168.

[86] Lasenby A N, Doran C J L. Closed Universes, de Sitter Space and Inflation, Submitted to: Phys. Rev. D. 2003.

[87] Mann Stephen, Dorst Leo. Geometric Algebra: a computational framework for geometrical applications (part II: applications), IEEE Trans. Computer Graphics and Applications, 2002 (8).

[88] Sommer G. Applications of Geometric Algebra in Robot Vision, In Proceeding of RIMS Symposium, Innovative Teaching in Mathematics with Geometric Algebra.

[89] Zaharia Marius. Finding the Diagram by Geometric Algebra Means, Computational Symbolic Geometry. Invariant Methods in Discrete and Computational Geometry, 1994: 107 ~139.

[90] Mourrain B. Enumeration Problems in Geometry, Robotics and Vision. MEGA, 1994. Also in: Algorithm in Algebraic Geometry and Applications, 1996 (143): 285 ~ 306.

[91] Mourrain B. New Aspects of Geometrical Calculus with Invariants. Adances in Mathematics, Also appear in MEGA 91, 1991.

[92] David Cox, Little John, O'shea Donal Ideals Varieties and Algorithms (znd ed) . Springer, 1991.

[93] Geddes K O, Czapor S R, Labahn G. Algorithms for Computer Algebra. Kluwer Academic Publishers, 1992.

[94] Franz Winkler. Polynomial Algorithms in Computer Algebra. preprint Edition, 1996.

[95] Bhubaneswar Mishra. Algorithmic for Computer Algebra. Springer-Verlag, 1993.

[96] A Dictionary for Elementary Geometry. Shanghai Educational Press, 1984.

[97] C. F. Adler. Modern Geometry. McGraw-Hill Book Company Inc. 1958.